UNIVERSIDAD CENTRAL DE VENEZUELA

FACULTAD DE CIENCIAS

ESCUELA DE COMPUTACION

UNA HERRAMIENTA BASADA EN TECNOLOGIA PALM PILOT PARA EL SISTEMA INDUSTRIAL DE CREDITO: ANALISIS DE FACTIBILIDAD Y DISEÑO DE UNA INTERFAZ DE COMUNICACIÓN.

Trabajo Especial de Grado presentado ante

la ilustre Universidad Central de Venezuela,

por el Bachiller:

Angel Enrique Gascón Garcia

C.I. 11.590.838

Para optar al título de

LICENCIADO EN COMPUTACION

Tutor: Rina Suros

CARACAS, SEPTIEMBRE DE 2009

Universidad Central de Venezuela

Facultas de Ciencias

Escuela de Computación

RESUMEN

El banco de la Gente Emprendedora tiene planteado dentro de sus estrategias organizacional la implantación de un sistema que apoye la gestión de la asistente administrativo y el asesor de negocio, automatizando el proceso de transcripción de información de las solicitudes de crédito de los clientes y mejorando los procesos de obtención de información del estatus de la cartera de crédito para el asesor de negocio; el mejoramiento de ambos procesos aumentaría la productividad de ambas áreas, lo que se traduciría en un ofrecimiento adecuado del servicio que se le presta a los clientes y el cumplimiento del eslogan de la organización “créditos rápidos y sin tanto papeleo”.

Otra ventaja para el desarrollo de un sistema de esta envergadura es lo relacionado al ahorro de recursos materiales tales como papel, tinta, consumibles, espacio físico de equipos destinados para asesores de negocio, etc.

Por lo antes expuesto es que se está desarrollando un sistema basado en tecnología Pal Pilot (Dispositivos móviles) para la captura de información en campo de las solicitudes de crédito de clientes, que a través de su respectiva interfaz podrá llevar dicha información al Core principal de la organización eliminando las transcripciones innecesarias, el gasto excesivo de papel, uso de formularios, consumibles, etc. y eliminara en gran medida el uso y tiempo requerido de la fuerza de venta en un computador personal, ya que la mayoría de la información requerida estará disponible en su dispositivo de bolsillo (Palm Pilot) a través de las distintas sincronizaciones con la interfaz, mejorando así eficientemente el servicio que se le presta a los clientes de esta prestigiosa organización Financiera.

INTRODUCCIÓN

En las ultimas décadas las organizaciones en general han tenido la necesidad de obtener información de manera amigable, rápida, sencilla y veraz, originando un gran crecimiento y desarrollo tecnológico; lo que se ha traducido en la adquisición de Tecnología Actualizada (Hardware y Software) y el mejoramiento de los sistemas de información, para brindar un servicio acorde y adecuado a este mundo tan cambiante y exigente. Todo este desarrollo tecnológico ha provocado una gran aceptación del poder analítico de las técnicas de computación y hemos visto el beneficio obtenido en los tiempos de procesamiento de la información.

Para tener una idea de estos avances tecnológicos, bastaría con nombrar el uso en las grandes empresas de las aplicaciones de Internet y microcomputadoras de bolsillo Palm Pilot, que brindan a éstas nuevas y excitantes posibilidades. Hace unos años aparecieron las primeras aplicaciones de telefonía en Internet que permitían establecer conversaciones telefónicas, incluyendo llamadas a larga distancia, utilizando Internet como transporte de la voz, más recientemente las aplicaciones de Internet que permiten recibir audio en tiempo real y hoy en día nos deslumbran con el Virtual Reality Modeling Language, más conocido como VMRL, el cual, es un lenguaje concebido para diseñar Mundos Virtuales distribuidos en la red de Internet, que permite a los usuarios navegar por estos mundos, abrir puertas, utilizar objetos y compartir elementos haciendo, de éste un mundo con enfoque más humano. **[VALZA]** En muchos sentidos el VRML es realmente muy parecido al HTML, que es el lenguaje que se utiliza para la descripción de páginas en el World Wide Web **[MAS96]**, pero con la diferencia de que el VRML permite que los diseños de mundos virtuales puedan ser visualizados utilizando diferentes tipos de computadores sin ninguna adaptación o esfuerzo especial **[ACTIV].** Imaginemos tener una aplicación completa en una computadora de bolsillo y poder obtener información en tiempo real al momento exacto en que la necesite y sin tener que trasladarse para ello, objetivo que es alcanzado con las aplicaciones, desarrolladas por terceros, en los dispositivos **"Palm Pilot"**.

En este sentido, entre las organizaciones con más auge tecnológico de los últimos tiempos en Venezuela se encuentran las bancarias. Estas instituciones se apoyan en la tecnología para alcanzar rápidamente sus metas y objetivos. Para tener una idea más precisa de este punto podemos dar unos ejemplos sencillos de estos avances tecnológicos:

- Bolívar Banco, organización que desde sus comienzos ha dirigido su desarrollo tecnológico a los sistemas bancarios basados en tecnología WEB, ofreciendo una gran gama de servicios bancarios a través de su portal en la Internet.
- Banco de Venezuela, Banco Provincial, Banco del Caribe, etc; han dirigido más su desarrollo tecnológico a los Sistemas de Información para atención al cliente en agencia. Aunque esta organización posee tecnología WEB su desarrollo no ha sido tan notorio como el mencionado anteriormente.
- El Banco de la Gente Emprendedora (BanGente) tiene enfocado su desarrollo a mejorar la atención y el servicio que se le presta al cliente tanto en agencia como en campo (sitio donde se encuentre la microempresa). A diferencia de la mayoría de las instituciones bancarias, BanGente es una organización especializada en el otorgamiento de Créditos a Microempresarios (*target,* clientes no atendidos dentro de las organizaciones bancarias tradicionales - bancos universales-) y deja en un segundo plano los productos pasivos de la organización.

Por lo anteriormente planteado, la institución financiera BanGente tiene, entre sus principales objetivos, mejorar su Sistema de Información de Crédito (Sicredito) incorporando para esto la tecnología Palm Pilot, sobre la cual se pretende recoger toda la información del cliente, requerida por la organización. Este sistema Sicredito permite el análisis interactivo de la información del crédito del cliente y esta conformado por varios módulos que permiten darle continuidad y aprobación a los créditos en la organización, entre los cuales tenemos:

- *Módulo de Operaciones*, aquí se realiza la trascripción, documentación y liquidación de los instrumentos crediticios.
- *Módulo de Aprobación.* Este módulo permite a los integrantes del Comité de Aprobación de Crédito, verificar la información de los clientes, revisar los estados

financieros familiares y de la microempresa, aprobar y plasmar sus condiciones de aprobación.

- *Módulo de Operaciones Documentación Consultoría.* Este módulo permite a la consultoría jurídica verificar la documentación automática del sistema y corregir la información transcrita.
- *Módulo de Contraloría.* Permite al personal asignado para tal fin verificar la información generada en la contabilidad de los cierres del sistema.
- *Módulo de Reportes.* Permite visualizar los reportes solicitados y necesarios para verificar, controlar y analizar los créditos de los clientes, cartera activa, etc.
- *Módulo de administración.* Facilita al personal asignado llevar los controles necesarios para los permisos del sistema, asignación de parámetros y perfiles, cierres de agencias, etc.

Ahora bien, como podemos ver la creación de un nuevo módulo para Palm Pilot que permita realizar las sincronizaciones de información entre las Bases de Datos (BD) del dispositivo y de la organización, traería consigo el hecho de realizar una Reingeniería de Procesos, ya que modificaría la forma de trabajo del personal que se encarga de la trascripción de la información.

En este documento se expondrán tres capítulos, en el Primer Capítulo se describe el problema a solucionar, se presenta la justificación de la Investigación, junto con sus objetivos generales y objetivos específicos, así como los alcances y limitaciones del proyecto. En el Segundo Capítulo se expondrán los conceptos básicos que serán de ayuda en la propuesta de la solución del problema. En el Tercer Capitulo se expondrá la propuesta para la solución del problema.

CAPÍTULO I

PROBLEMA

Planteamiento del Problema

En toda organización bancaria existe **la necesidad de captación progresiva de clientes** y por ende el hecho de prestarles un mejor servicio. El personal adjunto a la Gerencia Comercial de la empresa Banco de la Gente Emprendedora (BanGente C.A.) es el recurso humano encargado de llevar con éxito tal labor. Lograr esta captación de clientes lleva implícita una ardua labor publicitaria, pero a su vez es necesario prestarles **un servicio rápido y eficaz**, lo cual, no solo lleva a una captación segura sino a la retención de esta vital fuente de recursos para la empresa, por medio de herramientas adecuadas y de alto poder tecnológico que le permitan promover su desempeño.

Apoyándose en el criterio de prestar **un servicio rápido y eficaz** para sus clientes, BanGente a través de la Gerencia Comercial y la Gerencia de Tecnología, se ha propuesto considerar un sistema basado en la tecnología PalmPilot, junto con la interfase de comunicación con su Sistema Industrial de Crédito (Sicredito) que le permita a su personal dar servicios de asesoría y administración bancaria lo más eficientemente posible. Así mismo, debe permitir establecer el control primario del riesgo que representan los clientes para la organización en el proceso de interfase con las Bases de Datos de Riesgo del Banco, como está establecido por sus reglamentos internos, el de la Asociación Bancaria de Venezuela y la Superintendencia de Bancos (SUDEBAN).

Las actividades de asesoría prestadas por el personal adjunto a la Gerencia Comercial, que llamaremos de aquí en adelante *Asesores de Negocios*, son llevadas manualmente, generando una gran cantidad de trabajo innecesario. Este proceso de generación de solicitud de crédito lleva consigo los siguientes pasos:

- Levantamiento de información en campo (lugar donde esta ubicada la microempresa) por parte de los Asesores de Negocios.

- Trascripción de la información de campo en las plantillas de recolección de datos financieros implantadas por la organización bancaria, la cual debe ser plasmada en letra legible.
- Consultas de los clientes en los sistemas de riesgo de la organización bancaria para validar el riesgo de sus clientes.
- Trascripción de la información del cliente al sistema Sicredito, la cual es llevada por el personal administrativo de la misma área comercial; este personal es denominado *Asistentes Administrativos*.

Todo esto implica una sobrecarga de trabajo para el personal de la Gerencia Comercial y los servicios administrativos que se brindan no contemplan los mejores estándares debido a que se hace tedioso, y a veces difícil, el manejo de la información del principal producto del banco: los créditos. Además se desperdicia gran cantidad de papel de los formatos utilizados por los *Asesores de Negocios* (aumentando los costos de la organización). Aparte del volumen físico que este representa cuando el *Asesor de Negocios* tiene que hacer el levantamiento de información del cliente en campo.

Parte de la visión de BanGente es crecer y desarrollarse como una entidad financiera sólida al servicio de los microempresarios, lo que le exige que se preocupe por el recurso más valioso e importante con el que cuenta, y que representa la principal fuente de ingreso para la organización, por esta razón, es de vital importancia que al cliente siempre se le den los servicios más rápidos y eficientes para su confort y beneficio, pero siempre con la misma calidad que la ha caracterizado.

Debido a lo antes expuesto, la organización BanGente se ha propuesto realizar un estudio de factibilidad de los principales sistemas existentes, para el levantamiento de información en Palm Pilot del micro crédito empresarial y desarrollar una interfase que funcione como sistema de comunicación entre las Bases de Datos (BD) del sistema del dispositivo PalmPilot, la Base de Datos del Sistema Industrial de Crédito y la Base de Datos central de Riesgo del banco; lo cual le permitirá a la organización ahorrar tiempo, papel y prestar un servicio de forma más eficiente.

Para lograr su propósito de forma eficiente y prever un plan de expansión a todas las filiales del país de la forma más efectiva posible, BanGente ha buscado apoyo técnico en la Universidad Central de Venezuela (UCV), en particular con miembros del Centro de

Computación Paralela y Distribuida (CCPD), de esta forma se garantiza la transferencia tecnológica a corto plazo.

Ahora bien, durante el desarrollo de este trabajo de grado se realizó la búsqueda y el estudio de factibilidad del aplicativo para el levantamiento de información en Palm Pilot del Micro Crédito Empresarial, con lo cual obtuvimos dos aplicaciones que cumplían con las normas y expectativas primarias de la organización, una de ellas desarrollada por Acción Internacional Perú y la otra por Acción Internacional Colombia (cabe destacar que Acción Internacional es pionera en el estudio de las Micro Finanzas en América Latina). De los estudios realizados a los aplicativos y tomando en cuenta las capacidades físicas actuales de la organización, se hizo la escogencia del aplicativo desarrollado por Acción Internacional de Colombia ya que presenta una serie de ventajas. A continuación se presentaran las características y ventajas fundamentales de ambos aplicativos.

Aplicativo para Palm Pilot de Acción Internacional Perú:

Lenguaje de Programación: Codewarrior para Palm con Sistemas Operativos OS.

Conductor de Datos de Palm: Archivo creado por el programador en lenguaje Visual C++ con extensión .dll, el cual tiene la funcionalidad de detectar que registros de las tablas de la aplicación en Palm han sido modificados y trasladarlos al PC como archivos de texto que contienen únicamente aquellos registros modificados entre cada sincronización.

Ventajas

- Los tiempos de sincronización son más cortos ya que estamos trabajando únicamente sobre registros de tablas modificadas entre cada sincronización y esa información es la que se trasladaría de la Palm al PC.

Desventajas

- El manejo de las transacciones en las sincronizaciones se consideran abiertas, ya que dicha información debe ser guardada por registro en la BD en tablas especificas para ello y solo deberán pasar al Sicredito una vez que se haya completado el ciclo completo de la solicitud del cliente en la Palm, lo cual puede resultar inseguro ya que la información podría no ser manejada exclusivamente por el asesor.

- Se ve la impetuosa necesidad de tener espacio físico de almacenamiento en las BD para poder almacenar la información parcial de las solicitudes de los clientes de los diferentes asesores mientras estas cumplen el ciclo completo de levantamiento de información en las Palm´s.

Aplicativo para Palm Pilot de Acción Internacional Colombia:

Lenguaje de Programación: Satellite Form 3.1 para Palm con Sistemas Operativos OS.

Conductor de Datos de Palm: archivo Active propio del lenguaje Satellite de extensión .ocx, con funciones para controlar los diferentes estados de la transmisión de datos entre la Palm y el Pc; las tablas de las BD de la Palm son depositadas en el Pc como archivos de tablas de Visual Basic (extensiones .dbf). **[PALM99]**

Ventajas

- El manejo de la transacción correspondiente a la solicitud de crédito introducida por el asesor en la Palm se realizará de manera cerrada, es decir, cómo se están manejando las tablas completas de la aplicación, únicamente se modificará o trasladará información al Sicredito una vez que la solicitud del cliente en Palm ha cumplido determinados estados.
- Es posible realizar respaldos detallados de la aplicación Palm en cada sincronización de la información del asesor con simplemente resguardar las tablas al momento de las sincronizaciones.
- Como las transacciones son de tipo cerrada, no se necesita espacio físico en disco adicional para guardar la información de las solicitudes de clientes que aún se encuentren en estados intermedios en la Palm.
- La programación bajo el lenguaje Satellite Form es sencilla y no necesita de un entrenamiento especializado para capacitar un programador de la organización.

Desventajas

- Dependiendo de las políticas y criterios utilizados para el manejo de la información de los asesores en las sincronizaciones estas podrían realizarse en un tiempo mayor que el aplicativo señalado anteriormente.

Justificación de la Investigación

La Gerencia Comercial de la organización requiere de una solución eficaz, segura e integrada que permita desarrollar, almacenar, administrar y entregar el valioso contenido de la información de sus clientes, logrando simplificar distintas tareas administrativas que implican procesos con ciertos trámites complejos y de mucho papeleo.

El costo de la infraestructura de la red, la ubicación geográficamente dispersa de los clientes a los que visitan los asesores de negocios y la complejidad del contenido de la información que ellos manejan, requieren una administración más eficiente e inteligente.

El manejo de la información mediante un dispositivo Palm y su interfase con el sistema principal del banco permite agilizar los trámites de crédito, evitando pérdidas de tiempo y posibles errores de trascripción cometidos por el asistente Administrativo, brindando así un mejor servicio al cliente; al mismo tiempo ayuda a la organización a minimizar el consumo de papel al evitar que los asesores de negocios tengan que llevar esta información de forma manual.

El asesor de negocios podrá obtener de manera rápida y sencilla, la información referente a su cartera de morosidad, las posibles renovaciones de crédito en el mes por venir y los vencimientos de las cuotas de sus clientes para realizar la cobranza, ya que con cada sincronización que este realice, dicha información deberá actualizarse en el sistema de manejo de información del dispositivo Palm Pilot

Además permitirá fortalecer los nexos Institucionales con la Escuela Computación de la UCV, contribuyendo a que otros estudiantes en el futuro puedan utilizar esta investigación como base para posteriores investigaciones sobre sistemas basados en tecnología Palm Pilot y su uso en la micro finanzas.

OBJETIVOS DE LA INVESTIGACIÓN

Objetivo General

Diseñar una propuesta de un sistema basado en tecnología Palm Pilot y su interfase con el principal Sistema de Micro Finanzas de la organización; garantizando la seguridad de los datos involucrados y que logre integrar, de manera automatizada, todos los procesos manuales del personal de Asesores de Negocios y Asistentes Administrativos de la Organización BanGente.

Objetivos Específicos

- Realizar un estudio de los sistemas de información existentes en el mercado, basados en la tecnología Palm Pilot, dedicados al levantamiento de información de Micro Créditos o Micro Finanzas.
- Analizar los procesos críticos del área de la Gerencia Comercial de la Organización BanGente.
- Diseñar y desarrollar un sistema o interfaz que permita la comunicación entre los Sistemas Manejadores de Bases de Datos (SMBD) del Sistema de Micro Finanzas para Palm Pilot y el Sistema Industrial de Crédito (Sicredito).
- Realizar charlas e instructivos que permitan minimizar la resistencia al cambio de la tecnología que siempre se espera en este tipo de proyectos.

VENTAJAS ADICIONALES

Realizar el estudio de factibilidad, diseñar e implementar una interfaz para:

1. Ofrecer servicios de una forma más eficiente, permitiendo establecer un control de calidad de la atención que se le brinda al cliente, lo que le permitirá a la organización promover el buen desempeño de su personal adjunto a la Gerencia Comercial.
2. Establecer responsabilidades sobre el manejo de los datos de la información de los clientes, ya que actualmente esta responsabilidad es compartida entre el *Asesor de Negocios y el Asistente Administrativo*, pero con el manejo de la información en Palm sería el *Asesor de Negocios* quien trabajaría y depositaria la información en el sistema Sicredito sin intervención directa de los Asistentes Administrativos; aunque estos seguidamente puedan modificarla.
3. Disminuir el consumo de papel y el tiempo empleado en actividades rutinarias de trascripción de información, permitiendo así una reducción notable en los costos de la organización.

LIMITACIONES

Para desarrollar esta propuesta se presentan las siguientes limitaciones:

Los dispositivos o computadoras de bolsillo a utilizar son las Palm Pilot; una vez decidido el sistema que se va a utilizar en dichos dispositivos para apoyar el levantamiento de información de los Micro Créditos de la Organización, este solo podrá ser modificado en código por el personal externo de la empresa del cual se adquiera. Por ser un sistema propietario, dicha empresa se encargara de acondicionarlo a las reglas impuestas para las Micro Finanzas en Venezuela y hará prevalecer su garantía por dicho producto.

La interfaz entre los sistemas tiene que ser programado en el lenguaje PowerBuilder 9.0, apoyándose en el Sistema Manejador de Bases de Datos SQL Server 2005. Estas condiciones son impuestas por la Gerencia de Tecnología de la organización BanGente, ya que esto forma parte de su plataforma tecnológica.

CAPÍTULO II

MARCO TEÓRICO

Levantamiento de Información Institucional

Antecedentes de Sicredito

En visitas realizadas a la Gerencia de Tecnología de la Organización BanGente se pudo constatar que el Sistema de Información de Crédito (Sicredito) es un módulo crediticio que proviene originariamente del Sistema de Información Bancaria (SiBanca) creado por la Compañía Enlace Informático, que es una dependencia del Banco Solidario del Ecuador. Dicho sistema (Sicredito) es comprado por BanGente en Febrero de 1998 y la empresa Enlace Informático se compromete a la adaptación de dicho producto para que funcione como la principal herramienta de Control, Administración, Supervisión y Aprobación de créditos de la Organización BanGente. Siendo a comienzos del año 2001 que el banco decide prescindir de los servicios de Enlace Informático como compañía de Outsourcing y tomar el control de la programación de su sistema Sicredito, robusteciendo aun más dicha herramienta con la actualización de tecnología de punta, como el desarrollo del presente proyecto. Dicho sistema se caracteriza principalmente por ser completamente distribuido, es decir, tanto la aplicación como las bases de datos del Sistema Sicredito se encuentran almacenados de forma independiente en cada servidor de las agencias del banco, con lo cual se obtiene una clara ventaja, ya que las agencias del banco no dependen de servicios de enlaces de comunicación para que funcionen sus sistemas. Pero, si crea una alta dependencia de dichos enlaces a nivel gerencial para los requerimiento de información entre agencias y aprobación de créditos.

Bases Teóricas

Organización

Esta palabra proviene de "organismo", que según el diccionario **[OLI92]**, significa crear una estructura con partes integradas de tal forma que la relación de una y otra está gobernada por su relación con el todo. Un organismo, o los resultados de la organización, está conformado por dos ingredientes básicos que se identifican: partes y relaciones.

"Para que un analista de sistemas pueda analizar y diseñar sistemas adecuados de información, necesita entender su organización, pues la forma que tomen los sistemas dependerá, en gran medida, de la influencia de tres de sus principios organizacionales. Estos fundamentos organizacionales son: niveles de administración, el diseño de las organizaciones y un grupo de factores influyentes (como el estilo de liderazgo de quien toma las decisiones, la tecnología y las múltiples subculturas organizacionales)." **[KEN91]**

Todas las organizaciones, incluyendo la estudiada, son sistemas grandes integrados por subsistemas interrelacionados. "Dichos subsistemas tienden a verse afectados por los niveles de toma de decisión administrativa (operacional, administración media y dirección estratégica), los cuales inciden horizontalmente sobre el sistema organizacional." **[KEN91]**. Dicho sistema organizacional o diseño de la organización define su estructura, lo cual la obliga a tomar un modelo de tipo jerárquico y por consiguiente, también nos lleva a definir los niveles de responsabilidad y de autoridad dentro de la organización.

BanGente está integrada por pequeños sistemas interrelacionados (Contabilidad, Procesamiento de Datos, Gerencia Comercial, Operaciones, Recursos Humanos, etc.), que realizan funciones especializadas y que sin alguna de ellas no podría funcionar correctamente. Todas juntas conforman una eficaz entidad.

Según lo anteriormente expuesto, se puede decir que los tres niveles de control administrativo, las diferentes estructuras de la organización, el estilo de liderazgo, las consideraciones técnicas, la cultura de la organización y las interacciones humanas son factores que se interrelacionan en el sistema de la organización y tienen implicaciones en el análisis y diseño de los sistemas de información. El planteamiento y conocimiento de

estos factores permite, a los analistas de sistemas, mantener una amplia visión de la organización, y no un enfoque exclusivo sobre los sistemas de información para no descuidar los grandes aspectos de implantación del cambio dentro de la organización.

Diseño de la Organización

"Las organizaciones están diseñadas para cumplir, de la manera más efectiva y eficiente posible, con sus metas y objetivos. El diseño o la estructura tiene un claro propósito de que el trabajo rutinario se realice con el consumo mínimo de recursos, tale como personal, tiempo, materias primas e información." **[KEN91]**

El diseño de organización más común y que se identifica con facilidad es la estructura jerárquica, tal y como se observo en reunión realizada con la Gerencia de Recursos Humanos de la institución objeto de estudio, la cual nos facilito su estructura jerárquica (Figura 1):

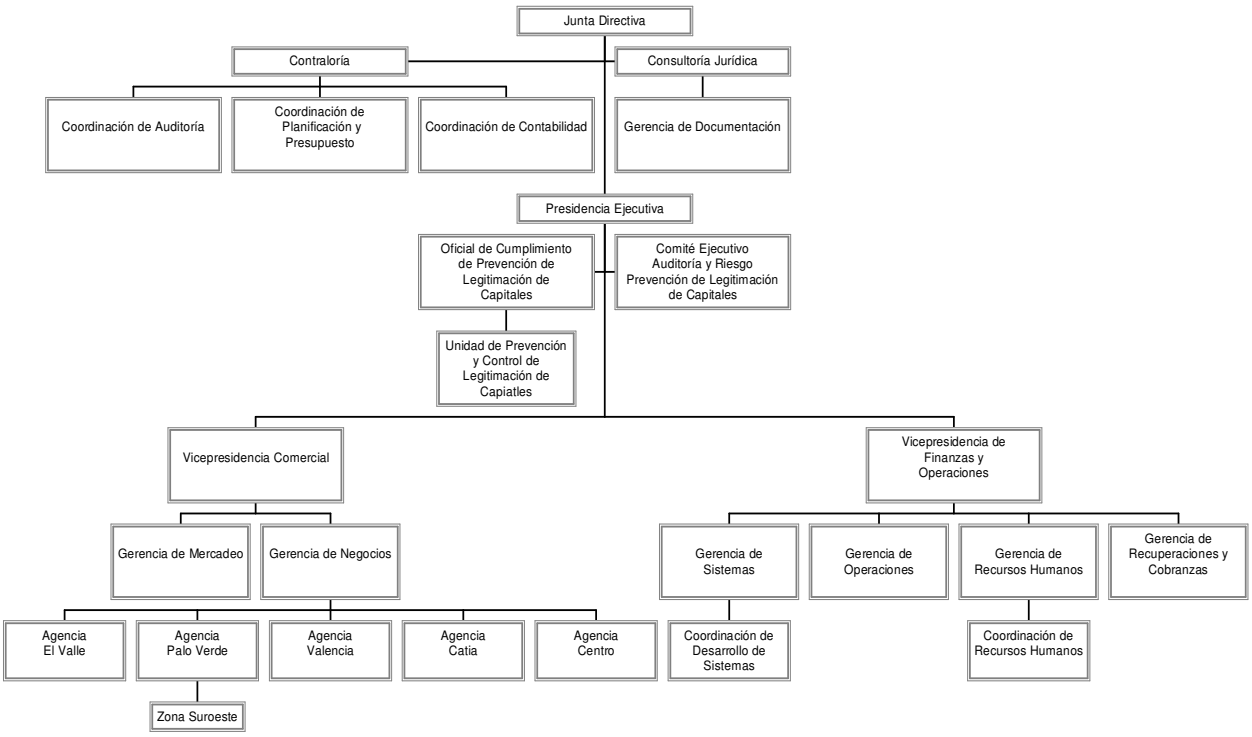


Figura 1. Organigrama estructural de la organización.

Como toda estructura jerárquica, está diseñada para cumplir las metas y objetivos trazados por la organización, como es el brindar eficazmente todos los servicios financieros, especialmente créditos, a las personas y empresas que realizan actividades productivas lícitas (parte de la misión de BanGente), mediante el empleo de diversos recursos (tecnológicos, factor humano, etc).

Determinación de Requerimientos

"La *determinación de requerimientos* es el estudio de un sistema para conocer cómo trabaja y dónde es necesario efectuar mejoras. Los estudios de sistemas dan como resultado una evaluación de la forma cómo trabajan los métodos empleados y si es necesario o posible realizar ajustes. Como se verá más adelante, estos estudios consideran métodos tanto basados en computadoras como manuales; es decir, no se circunscriben exclusivamente a estudios de cómputo.

Un *requerimiento* es una característica que debe incluirse en un nuevo sistema. Esta puede ser la inclusión de determinada forma para capturar o procesar datos, producir información, controlar una actividad de la empresa o brindar soporte a la gerencia. Es así como la determinación de requerimientos vincula el estudio de un sistema existente con la recopilación de detalles relacionados con él." **[SEN92]**

En nuestro caso especifico de investigación y análisis debemos realizar la determinación de requerimientos sobre el proceso de captura de datos que realizan los asesores de negocios de la organización, para llevar a cabo adecuadamente el estudio de factibilidad de los sistemas de levantamiento de información de Micro Crédito y dar las pautas de lo que será la adaptación del producto. Este mismo proceso de determinación de requerimientos nos va ayudar a definir como debe realizarse correctamente la interfase de comunicación entre el sistema del dispositivo Palm Pilot y el Sistema Sicredito. Debemos analizar de la misma manera como se realiza el proceso manual actual de carga de datos en el sistema, por parte de los asistentes administrativos. Todo esto con la única intención de comprender la situación claramente, ya que un analista de sistemas no tiene los mismos conocimientos, hechos y detalles que los usuarios y gerentes de las áreas involucradas.

Actividades de la determinación de requerimientos

Podemos ver la determinación de requerimientos por medio de tres actividades fundamentales, como son:

Anticipación de requerimientos

Este punto es referido a la experiencia que posee el analista con sistemas o áreas particulares cuyos ambientes y características son similares al que se esta analizando, lo que proveería algunos problemas, características y hasta requerimientos para el nuevo sistema.

"La anticipación de requerimientos puede ser una mezcla de bendiciones. Por un lado, la experiencia de estudios previos puede conducir a la investigación de áreas que no consideraría un analista novato. Tener las bases necesarias para saber qué preguntar o qué aspectos investigar puede ser un beneficio sustancial para la organización.

Por otra parte, si se introducen sesgos o atajos al conducir la investigación, entonces es muy probable que la anticipación de requerimientos se convierta en un problema. Por tanto, siempre deben darse lineamientos para estructurar una investigación alrededor de cuestiones básicas." **[SEN92]**

Investigación de requerimientos

Es una de las actividades más importantes del análisis de los sistemas y depende en gran medida de las técnicas utilizadas por el analista para encontrar datos o hechos, que pueden incluir métodos para documentar y describir las características del sistema.

Especificaciones de requerimientos

"Los datos obtenidos durante la recopilación de hechos se analizan para determinar las especificaciones de los requerimientos, es decir la descripción de las características del nuevo sistema. Esta actividad tiene tres partes relacionadas entre si:

- *Análisis de datos basados en hechos reales*: se examinan los datos recopilados durante el estudio, incluidos en la documentación de flujo de datos y análisis de

decisiones, para examinar el grado de desempeño del sistema y si cumple con las demandas de la organización.

- *Identificación de requerimientos esenciales*: características que deben incluirse en el nuevo sistema y que van desde detalles de operación hasta criterios de desempeño.
- *Selección de estrategias para satisfacer los requerimientos*: estos son métodos que serán utilizados para alcanzar los requerimientos establecidos y seleccionados. Estos forman la base para el diseño de sistemas, los cuales deben cumplir con la especificación de requerimientos."

[SEN92].

Técnicas para encontrar hechos

Dichas técnicas son consideradas métodos específicos utilizados para reunir datos relacionados con los requerimientos los cuales podemos combinar, para estar seguros de llevar a cabo una investigación amplia y exacta. Dichas técnicas podemos definirlas como:

- **Entrevistas**

"Los analistas emplean la entrevista para reunir información proveniente de personas o de grupos. Por lo común, los entrevistados son usuarios de los sistemas existentes o usuarios en potencia del sistema propuesto. En algunos casos, los entrevistados son gerentes o empleados que proporcionan datos para el sistema propuesto o que serán afectados por él. Aunque algunos analistas prefieren la entrevista sobre otras técnicas, esta no siempre es la mejor fuente de datos sobre la aplicación. Dado que la entrevista requiere de tiempo, es necesario utilizar otros métodos para obtener la información necesaria para conducir la investigación." **[SEN92].**

En este método podemos ver que puede ser de gran ayuda para recolectar información de personas que no pueden comunicarse adecuadamente por escrito o de aquellas que no poseen el tiempo necesario para llenar cuestionarios; así como también

resulta ser la mejor fuente de información cualitativa, donde se expresan opiniones, políticas, descripciones subjetivas de actividades y problemas.

"Las entrevistas pueden clasificarse con estructuradas y no estructuradas. Las entrevistas no estructuradas utilizan un formato pregunta-respuesta y son apropiadas cuando el analista desea adquirir información general acerca de un sistema. Este formato anima a los entrevistados a compartir sus sentimientos, ideas y creencias. Por otro lado, las entrevistas estructuradas utilizan preguntas estándar en un formato de repuesta abierta o cerrada. El primero permite que el entrevistado dé respuesta a las preguntas con sus propias palabras; el segundo utiliza un conjunto anticipado de respuestas." **[SEN92].**

- **Cuestionarios**

El uso de esta técnica nos puede proporcionar en cierta medida datos más confiables que otras técnicas utilizadas, ya que su amplia distribución asegura el anonimato de los encuestados proporcionándonos respuestas más honestas o respuestas limitadas debido a:

1. El encuestado no tiene la disposición de llenar fielmente el cuestionario.
2. No es posible observar las expresiones o reacciones de los encuestados.

"Con frecuencia los analistas utilizan cuestionarios abiertos para descubrir sentimientos, opiniones y experiencias generales o para explorar un proceso o problema. Los cuestionarios cerrados controlan el marco de referencia al presentar a los encuestados respuestas específicas para escoger. Este formato es apropiado para obtener información basada en hechos reales." **[SEN92]**.

- **Revisión de los Registros**

Esta técnica examina la información relacionada con el sistema y los usuarios que contienen los registros de la organización, lo cual podría darnos la base para comparar las operaciones que actualmente se manejan en la organización.

"Los registros incluyen manuales de políticas, reglamentos y procedimientos estándares de operación utilizados por la mayor parte de las organizaciones como guías para los gerentes y empleados. Estos registros no indican la forma en la que se desarrollan las actividades en la realidad, donde se encuentra todo el poder de la toma de decisiones, o

cómo se realizan todas las tareas. Sin embargo pueden ser de gran ayuda para el analista en su afán de comprender el sistema al familiarizarlo con aquellas operaciones que necesitan apoyo y con las relaciones formales dentro de la organización." **[SEN92]**.

- **Observación**

"Por medio de la observación el analista obtiene información de primera mano sobre la forma en que se efectúan las actividades. Este método es más útil cuando el analista necesita observar, por un lado, la forma en que se manejan los documentos y se llevan a cabo los procesos y, por otro, si se siguen todos los pasos especificados. Los observadores experimentados saben qué buscar y cómo evaluar la significancia de lo que observan." **[SEN92]**

Reingeniería

"Una breve definición de Reingeniería podría ser que significa "Empezar de nuevo". Pero en si la reingeniería, formalmente es la revisión fundamental y el rediseño radical de procesos para alcanzar mejoras criticas y contemporáneas de rendimiento, tales como costo, calidad, servicio y rapidez." **[INFOR]**

En tal sentido la reingeniería es el proceso de examinar los sistemas y procesos existentes y/o modificar éstos para:

- Actualizar la tecnología.
- Capturar los componentes de información en un almacenamiento adecuado.
- Incrementar la productividad de la organización.
- Mejorar la actualización y mantenimiento de las Bases de Datos y aplicaciones existentes

Reingeniería para mejorar la Rentabilidad de la Empresa

Uno de los mayores problemas al analizar la rentabilidad de las empresas es ver cómo esta se ha reducido y que este problema no ocurre necesariamente por la

disminución de las ventas. Cuando se está en este tipo de problemas una de las soluciones más efectivas es planificar y aplicar cambios, lo que hoy se llama *"Reingeniería"*.

Para realizar Reingeniería no solo hay que conocer la empresa, sino también el mercado, el mundo, la economía, la tecnología los competidores y la política del país. Para explicar la complejidad de la reingeniería hay que considerar muchos detalles y datos. Algunos de ellos son:

- Para ganar más dinero no hay que tocar los precios sino manejar los costos (hacia abajo por supuesto).
- Sobretodo en épocas de crisis, ciertos costos no hay que considerarlos gastos sino inversión y no se deben tocar. Hasta hay casos en que es necesario aumentarlos.
- Hay que analizar los ingresos y egresos área por área y ver que sección es más eficiente, que productos son estrella y cuales no.
- Cuando se transita un ciclo económico negativo o de recesión es cuando se toman decisiones estratégicas que en los ciclos buenos se evitan.
- No se puede olvidar motivar al personal, a los clientes y a los proveedores. De nada sirve que se haga una importante inversión en tecnología para un área de la empresa si no estudia hacerla en las otras y de la mano con sus proveedores y clientes. Además, de esta manera se pueden reducir costos de implementación.
- Hay que recordar que la confianza se logra paso a paso, o sea lentamente, y nunca de un día para el otro o por tomar una determinada medida.
- Hay costos que son difíciles de medir, como por ejemplo la publicidad, pero son necesarios y se pueden analizar dentro del proyecto general.

"La crisis es un desafío. Si hay que aceptar que hoy en día la complejidad es mayor, que la globalización antes no existía y que la tecnología hace aportes que modifican las estructuras de costos en forma vertiginosa y geométrica. Hasta se ha llegado al limite de que las empresas pierden mucho dinero por no saber que existe tecnología adecuada para reemplazar el modelo que utilizan por otro tecnificado y de un costo mucho menor." **[GON00]**

A continuación se describen las actividades a realizar para la especificación de los requerimientos de los usuarios, luego se presenta el Método OOSE **[JAC96]**, el cual es un método orientado a objeto para el desarrollo de software y el UML (Unified Modeling Language), que es una especificación de notación orientada a objetos.

Especificación de los Requerimientos

La especificación de los requerimientos persigue establecer claramente las expectativas de los usuarios con respecto al producto final. Esta fase proporciona suficientes recursos para explotar el problema de la organización y establecer una visión de la solución.

Se delimita el sistema y se define su funcionalidad a través de las siguientes actividades, como se expone en **[SEN92]**:

- **Determinar el perfil de la organización:** se busca obtener información clave de la organización que sea de utilidad en el proceso de análisis y diseño de la aplicación a desarrollar.
- **Determinar el perfil de los miembros de la organización:** se identifican los usuarios de la aplicación determinando sus habilidades y conocimiento, para obtener el perfil de los usuarios.
- **Analizar las tareas de los miembros de la organización:** se determinan que quieren hacer los usuarios y cómo llevan a cabo sus tareas.
- **Analizar el Ambiente de trabajo de los miembros de la organización:** se determina dónde llevan a cabo los usuarios sus tareas, se determinan características ambientales que pueden tener impacto en el trabajo de los usuarios.
- **Recopilar los requerimientos de la aplicación:** se determina que esperan los usuarios de la aplicación y su interfaz, además de determinar los requerimientos funcionales para el producto de software a desarrollar. Los requerimientos de los usuarios deben ayudar a determinar el diseño apropiado de la interfaz de usuario.

- **Validar los requerimientos finales expuestos por los miembros de la organización:** consiste en validar los requerimientos de los usuarios contra las tareas que llevan a cabo. Para ello deben manejar las percepciones y los requerimientos de los usuarios, de manera que no esperen más de lo que obtendrán. Al igual, se debe verificar que no se desarrolle algo más allá de lo que requieren los usuarios. Al revisar las tareas de los usuarios y los requerimientos, es posible determinar el impacto de los cambios y las medidas a tomar para facilitar la transición.

Método OOSE

El Método OOSE (Object Oriented Software Engineering), propuesto por Ivar Jacobson **[JAC96]**, es un método de desarrollo de software orientado a objeto en el que cumple un rol protagónico el enfoque de casos de usos (use case). Un caso de uso es una manera específica de usar el sistema ejecutando alguna parte de la funcionalidad. Cada caso constituye un curso de eventos comenzando por un actor y especifica la interacción que ocurre entre el actor y el sistema.

Para el método OOSE, el desarrollo de sistema se puede ver como un proceso de producir descripciones del modelo.

En esencia, el desarrollo de sistema consta de tres fases distintas que siguen una a la otra. Cada uno de estos procesos describe una actividad específica del sistema. En particular son tres los procesos que se llevan a cabo, estos son el proceso de análisis, proceso de construcción y el proceso de prueba Figura 2.

El desarrollo de un sistema se realiza entonces a través de estos tres grandes procesos y se concreta mediante la creación de cinco modelos:

- Modelo de Requerimientos
- Modelo de Análisis
- Modelo de Diseño
- Modelo de Implementación
- Modelo de Prueba

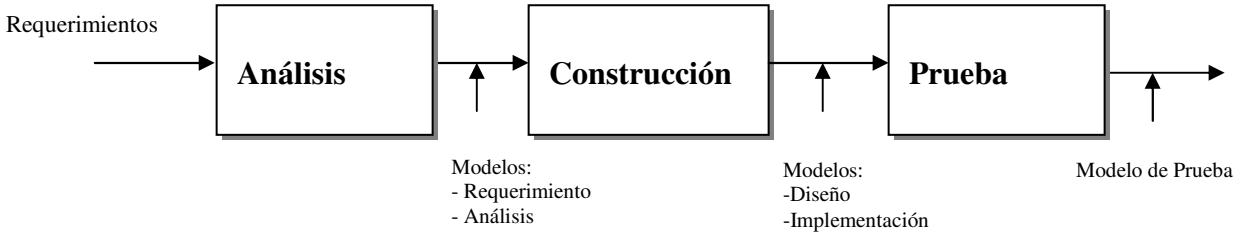


Figura 2. Procesos del Método OOSE

Modelo de Requerimiento

Webster´s Ninth New Collegiate Dictionary define un requerimiento como "algo requerido; algo querido o necesitado" El estándar IEEE 729 lo define como "(1) una condición o capacidad requerida por un usuario para resolver un problema o alcanzar un objetivo; (2) una condición o capacidad que se debe encontrar o que debe poseer un sistema para satisfacer un contrato, estándar, especificación, u otro documento formalmente impuesto."

Una Especificación de los Requerimientos del Software (SRS: Software Requirements Specification) es un documento que contiene una descripción completa de qué hace el software sin describir cómo lo hará.

El objetivo primordial del Modelo de Requerimientos es definir la funcionabilidad del sistema y establecer sus principales restricciones. Éstos describen todas las entradas y salidas para y desde el sistema así como la información acerca de cómo las entradas y las salidas se interrelacionan. Esta descripción de cómo entradas trazan en salidas se llama típicamente descripciones de Comportamiento o especificaciones operacionales y son realmente no trivial.

Los Requerimientos no de comportamientos es decir no funcionales definen los atributos del sistema cuando ejecuta su trabajo. Incluyen una descripción completa de los

niveles de eficacia, fiabilidad, garantía, mantenimiento, portabilidad, visibilidad, capacidad, y complacencia de las normas requeridos por el sistema.

Los requerimientos de comportamientos definen precisamente qué entradas son esperadas por el software, qué rendimientos serán generadas por el software, y los detalles de las relaciones que existe entre esas entradas y salidas. Para abreviar, los requerimientos de comportamientos describen todos los aspectos de interfaces entre el software y su medio ambiente (esto es, hardware, El modelo de requerimientos apunta para delimitar el sistema y definir la funcionalidad que el sistema debe ofrecer. El modelo de requerimientos será estructurado por el modelo de análisis, implementado por el modelo de diseño, y probado por el modelo de prueba.

El modelo de requerimientos consta de: el modelo de caso de uso, la descripción de la interfaz y el modelo del dominio del problema.

Modelo de Casos de Uso

El modelo de caso de uso especifica la funcionalidad que el sistema tiene que ofrecer desde una perspectiva del usuario y nosotros definimos qué debe ocurrir dentro del sistema. El modelo de caso de uso usa **actores** para representar roles que los usuarios pueden jugar, y **casos de uso** para representar qué los usuarios deben poder hacer con el sistema Figura 3. Cada caso de uso es un curso completo de eventos en el sistema, vistos desde la perspectiva del usuario.

Estos conceptos son simples y ayudan a definir qué existe fuera del sistema (actores), mientras que los casos de uso indican lo que este debe hacer.

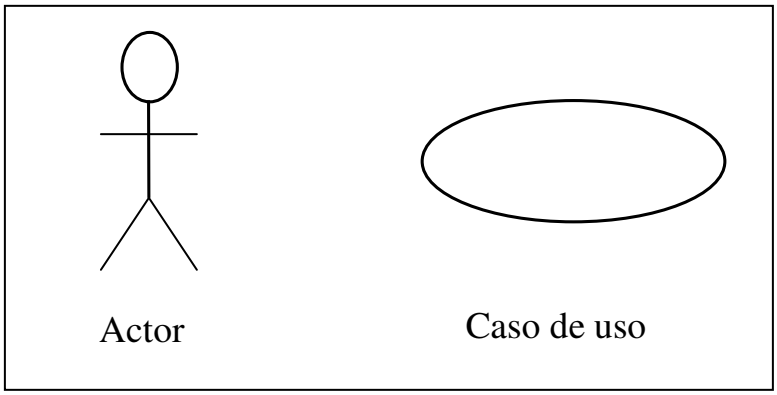


Figura 3. Notación del modelo de Caso de Uso

La realización de este modelo se logra en dos pasos:

- **Identificación de los actores**

Un actor es un tipo o categoría de usuario, y cuando un usuario hace algo él o ella actúa como una ocurrencia de este tipo. Una persona puede instanciarse (juega los roles de) varios actores diferentes. Los actores permiten modelar cualquier cosa externa al sistema que necesite intercambiar información con este, sea una persona u otro sistema. La distinción entre un actor y un usuario puede establecerse pensando en el actor como una clase.

Existen dos tipos de usuarios: los primarios y los secundarios. Los actores primarios son aquellos que trabajarán directamente con el sistema ejecutando sus tareas más importantes. Los actores secundarios existen solamente para que los actores primarios puedan usar el sistema (Ej. Un administrador de Base de Datos). La distinción entre un actor y un usuario puede establecerse pensando en le actor como una clase.

Por otra parte, cuando diferentes actores coinciden en algunos de sus roles puede definirse un actor abstracto al cual se le asignan los roles comunes. Así los actores concretos heredan los roles coincidentes del actor abstracto.

- **Definición de los Casos de Uso**

Los casos de uso representan lo que los actores pueden hacer con el sistema, es decir, los actores realizan un conjunto de casos de uso que definen la funcionalidad del sistema. Cada caso de uso constituye un curso completo de eventos comenzado por un actor y especifica la interacción que ocurre entre un actor y el sistema. Un caso de uso es así una sucesión especial de transacciones relacionadas ejecutada por un actor y el sistema en un diálogo. Los casos de uso recolectados especifican todas las maneras existentes de usar el sistema.

Para entender los casos de uso podemos ver sus descripciones como gráficas de transición de estado. Cada estímulo enviado entre un actor y el sistema ejecuta un cambio de estado en esta gráfica. Cada actor ejecutará varios casos de uso en el sistema.

Para identificar los casos de uso, podemos leer la especificación de los requerimientos desde una perspectiva del actor y llevamos discusiones con aquellos quienes actuarán como actores. Esto se realizará con la ayuda de preguntas, tales como:

¿Cuáles son las tareas principales de cada actor?

¿Tendrá el actor que leer / escribir / cambiar cualquier información del sistema?

¿Tendrá el actor que informar el sistema acerca de fuera de cambios?

¿Quiere el actor estar informado acerca de cambios inesperados?

Una clase del caso de uso es una descripción. Esta descripción especifica las transacciones del caso de uso. El conjunto de todas las descripciones del caso de usa especifica la funcionalidad completa del sistema. La Figura 4 ilustra un modelo de caso de uso.

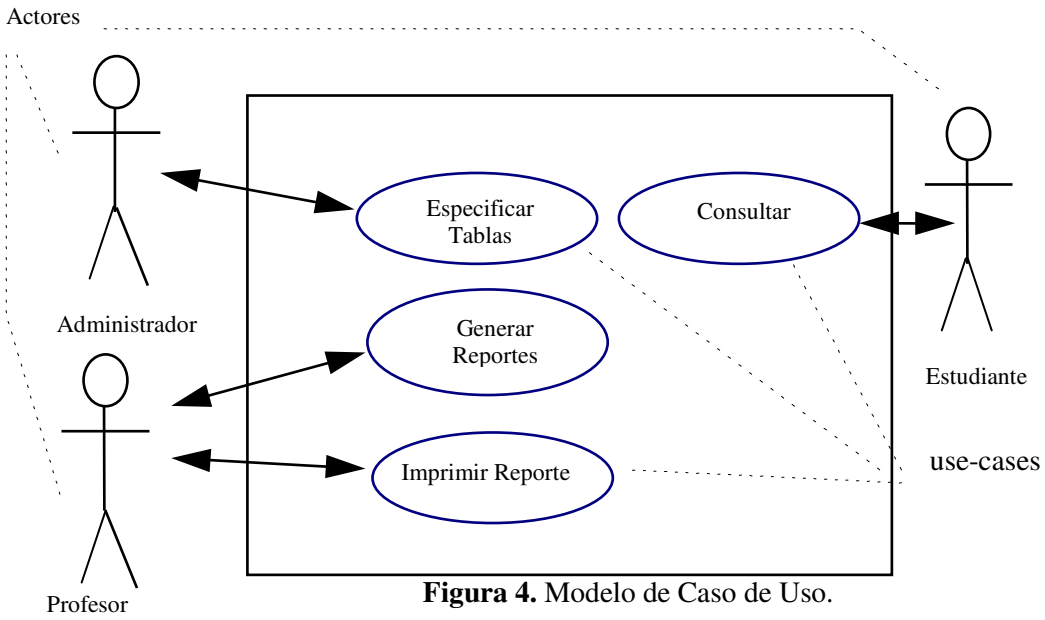


Figura 4. Modelo de Caso de Uso.

El modelo del sistema será un manejo de casos de uso.

Otra característica importante del modelo de los requerimientos es que podemos discutirlo con los usuarios y descubrir sus requerimientos y preferencias. Este modelo es fácil de entender y formular desde la perspectiva del usuario.

El refinamiento se logra desarrollando cada caso de uso, descomponiéndolo en varios casos de uso estructurados mediante las relaciones **[JAC96]**:

Use: es una asociación que relaciona cursos fuertemente acoplados que permiten aislar partes del caso de uso base. Se usa cuando:

Se quiere dividir un caso de uso base en casos de usos constitutivos.

Se quiere separar una parte del caso de uso base que por si misma constituye una funcionalidad importante dentro del caso de uso base.

Se identifica un caso de uso abstracto que luego será instanciado en otros casos de uso y pasa a ser un caso de uso concreto.

Extend: es una asociación que describe un curso alterno opcional (la extensión) de otro caso de uso (base). La extensión especifica cómo una descripción de caso de uso se inserta en, y así extiende, otra descripción del caso de uso. Se dibuja la asociación de la extensión con una flecha punteada ya que es una asociación de la clase. Las instancias de las asociaciones, se dibujan con líneas llenas. Se usa cuando se desea describir:

- Partes opcionales de un caso de uso base.
- Cursos alternativos que son ejecutados sólo en ciertos casos.
- Cursos separados que son ejecutados en ciertas condiciones.
- Situaciones donde pueden elegirse entre diferentes opciones.

Modelo de Interfaz

Para soportar el modelo de caso de uso es apropiado a menudo desarrollar bien las interfaces de los casos de uso. Aquí un prototipo de la interfaz del usuario es una herramienta perfecta. En esta manera podemos simular los casos de uso para los usuarios mostrando al usuario las vistas que ella o él verá cuando ejecuten el caso de uso en el sistema construido, como por ejemplo bosquejos sobre lo que el usuario verá en pantalla.

Al diseñar las interfaces hombre-máquina en las primeras etapas del desarrollo del sistema se intenta lograr que las interfaces sean un reflejo de la visión lógica que tienen los usuarios del sistema.

Modelos de los Objetos del Dominio del Problema

Este modelo da inicio al desarrollo de la visión lógica del sistema, esto es, la determinación de sus limitaciones y la definición de sus principales tareas. Esta basado en los objetos que pueden ser identificados en el dominio del problema los cuales tendrán en el sistema su correspondiente contraparte. El modelo de caso de uso controlará la formación de todos los otros modelos Figura 5.

Un modelo del dominio del problema se puede usar también cuando se hace la modelación de la empresa. Entonces es esencial para capturar todos los conceptos importantes y fundamentales en un modelo.

Esto significa que un modelo del dominio del problema, desarrollado con alguna clase de técnica, le servirá como una entrada muy sólida para el desarrollo de sistema.

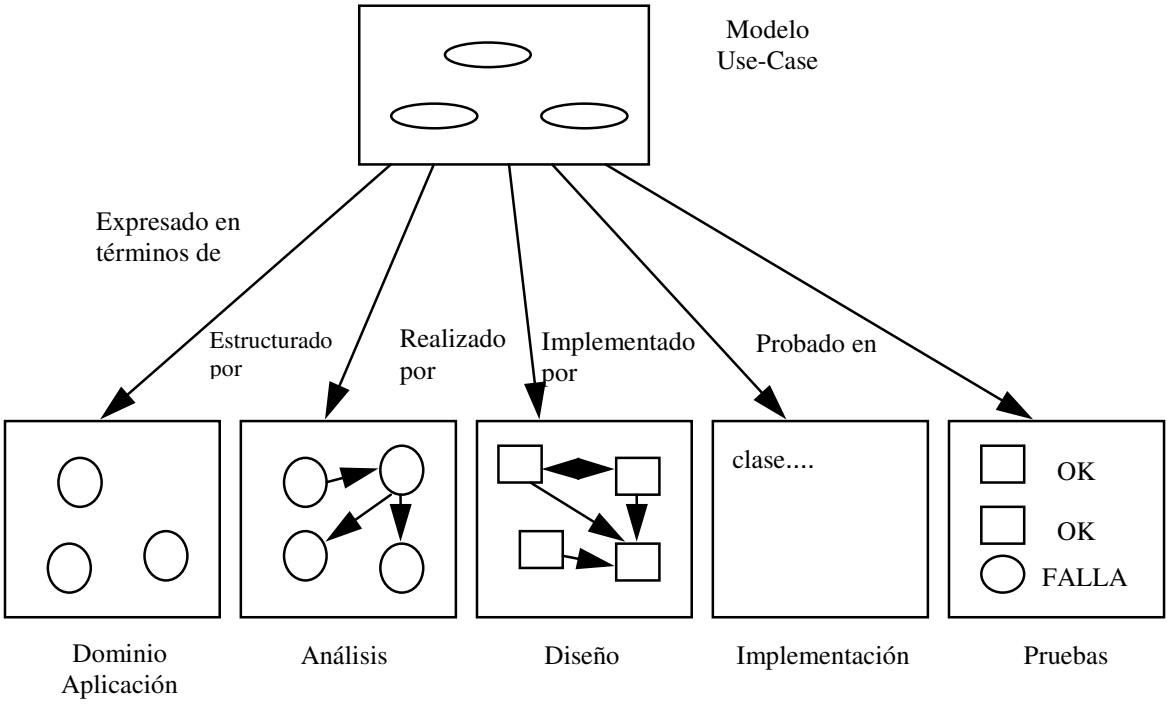


Figura 5. El modelo de caso de uso es usado para desarrollar otros modelos.

Esto se desarrolla en cooperación con el modelo del objeto del dominio y puede, en algunos casos donde el modelo del objeto del dominio se trabaja dentro de un modelo del objeto detallado, ser expresado en cuanto a objetos del dominio. El modelo se ajusta al medio ambiente de implementación real y más allá se refina en el modelo de diseño, usando los casos de uso para describir cómo los casos de uso fluyen sobre los objetos del diseño. Se implementarán los casos de uso entonces por la codificación fuente en el modelo de implementación. Finalmente los casos de uso nos darán una herramienta para la prueba del sistema, principalmente durante la prueba de integración. El modelo de caso

de uso nos dará también apoyo cuando escribamos manuales y otras instrucciones operacionales.

Modelo de Análisis

El modelo de requerimientos apunta para definir las limitaciones del sistema y especificar su conducta. Cuando el modelo de requerimientos ha sido desarrollado y aceptado por los usuarios del sistema o por el ordenador, podemos comenzar ha desarrollar el sistema real.

Habría sido posible usar el modelo de objeto desarrollado en el modelo de requerimientos como una base para la construcción real del sistema.

El modelo de análisis se construye para especificar objetos en este espacio de la información.

En el modelo de análisis describimos el sistema usando tres tipos diferentes de objeto: objetos interfaz, objetos entidad y objetos control Figura 6. Cada uno de estos objetos tiene su propósito propio y modelarán un aspecto específico del sistema. También usamos subsistemas para agrupar estos objetos en unidades manejables.

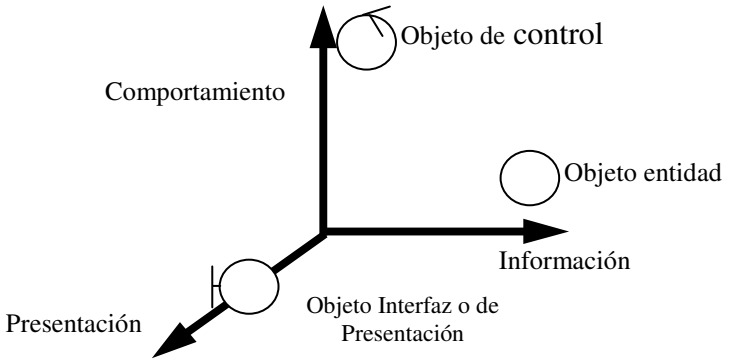


Figura 6. Las dimensiones y tipos de objetos del modelo de análisis.

Objetos Entidad

Este tipo de objeto permite modelar la información que será manipulada por el sistema y que se caracteriza por permanecer mucho tiempo en él e incluso, a la ejecución de los casos de uso. Toda conducta naturalmente acoplada a esta información debe ser colocada en el objeto entidad. Un ejemplo de un objeto entidad es una persona con sus datos asociados y conducta.

Para guardar información, los objetos usan atributos. A cada objeto podemos atar así varios atributos. Cada atributo tiene un tipo, que puede ser o un tipo de dato primitivo, tal como entero o string o un tipo de dato compuesto que es más complejo y el cual es específicamente definido. Un atributo se describe como una asociación.

La siguiente es una lista típica de operaciones que deben ser ofrecidas por un objeto entidad:

- Guardar y buscar información.
- La conducta debe cambiar si se cambia el objeto entidad
- Crear y eliminar el objeto entidad

Objetos de Interfaz

El objeto interfaz modela la conducta e información que son dependiente de la interfaz para el sistema. Así se coloca todo acerca de cualquier interfaz para el sistema en un objeto interfaz. Un ejemplo de un objeto interfaz es la funcionalidad de la interfaz usuario para solicitar información acerca de una persona.

La tarea de un objeto interfaz es traducir la entrada del actor al sistema en eventos en el sistema. Los objetos interfaz pueden, en otras palabras, describir la comunicación bidireccional entre el sistema y sus usuarios.

Los objetos interfaz son bastante simples de identificar. Tenemos por lo menos tres estrategias. O se identifican claramente de las descripciones de interfaz del sistema acompañadas del modelo de requerimientos, o comenzamos desde los actores, o podemos leer las descripciones del caso de uso y extraer la funcionalidad que está interfaz específica.

Cada actor concreto requiere su interfaz propia para su comunicación con el sistema.

Podemos diferenciar entre varias estrategias diferentes para asignar la funcionalidad:

- Control dominante de la Computación o control incluido es donde el lugar donde controlar la funcionalidad está dentro del sistema, esto es, en los objetos control y los objetos entidad. Aquí los objetos interfaz no tienen mucha funcionalidad.
- Control dominante del Diálogo es donde ponemos mucha funcionalidad de control en los objetos interfaz y estos objetos modelan mucho la funcionalidad del sistema. En este caso no tenemos muchos objetos control en el modelo. Esta estrategia es fácil para prototipo, pero incrementa la complejidad de las interfaces ya que diferentes niveles de abstracción se mezclan.
- Control mixto lugares de control en ambos lados permitiendo invocación de diálogo del lado computacional y viceversa. Esto ofrece más flexibilidad, pero requiere a los programadores estar más disciplinados para mantener independencia del diálogo.
- Control de Balanceo es donde separamos el control de tanto el diálogo como de la computación. El componente de control global, que típicamente es un objeto control, gobierna la secuencia entre invocaciones de diálogo y funciones computacional.

Objetos de Control

Los objetos de control permiten modelar el comportamiento que no puede ser asignado naturalmente a cualquiera otro objeto, por ejemplo, la conducta que consiste en operar en varios objetos entidad diferentes, hacer algunos cálculos y entonces retornar el

resultado a un objeto interfaz. Un ejemplo típico de un objeto control calcula impuestos usando varios factores diferentes.

Los objetos control típicamente actúan como cola que se une a los otros objetos para forman un caso de uso.

Los objetos control son normalmente encontrados directamente desde los casos de uso. En un proyecto preliminar asignaremos un objeto control para cada caso de uso abstracto y concreto.

Cada caso de uso normalmente envuelve objetos interfaz y objetos entidad. Así la conducta que queda después de los objetos interfaz y los objetos entidad se han asignado a su conducta en los objetos de control.

Subsistemas

Generalmente, son muchos los objetos que son definidos en un sistema dificultando que se logre una visión de conjuntos de éste. De aquí que, con frecuencia sea conveniente formar agrupaciones de objetos denominado subsistemas, cada uno de los cuales a su vez puede estar constituido por otros subsistemas.

La tarea de los subsistemas es empaquetar los objetos para que se reduzca la complejidad. Los subsistemas también trabajan como unidades de manipulación en la organización, por ejemplo para el desarrollo, mercadeo, ventas y entrega. Un subsistema puede ser una unidad del sistema compulsiva, pero puede ser también una unidad optativa.

El nivel más bajo de subsistema se ve como una unidad de cambio. Llamamos a estas unidades **paquetes de servicio** (service packages) y son considerados como unidades atómicas. La base de información de cada uno de estos paquetes son los distintos objetos del análisis.

La tarea de los subsistemas es empaquetar los objetos para que se reduzca la complejidad, pero esta división también debe estar basada en la funcionabilidad del sistema.

Modelo de Diseño

El modelo de Diseño se obtiene refinando el Modelo de Análisis. Este último, que fue construido asumiendo condiciones ideales, debe ser adaptado en el Modelo de diseño al ambiente de implementación.

Este modelo puede ser considerado como una formalización del espacio de análisis. Esto se logra incluyendo en este último otra dimensión: la del ambiente de implementación con la que queda definido entonces el **espacio de diseño.** La nueva dimensión es necesaria porque al introducir aspectos específicos de la implementación, (Ej. El uso de un manejador de Base de Datos) posiblemente se originaran cambios en la estructura del sistema.

En el modelo de Diseño se utiliza el concepto de **bloque** para describir la forma como debe organizarse el código. Los bloques son los objetos de diseño. Cada uno de los bloques intenta implementar un objeto del análisis, por lo tanto se tendrán: bloques entidad, bloques de interfaces y bloques de control: los bloques pueden ser considerados como una abstracción del sistema ya que estos constituirán agrupaciones de código fuente.

Podemos considerar el modelo de diseño como una formalización del espacio de análisis, donde ajustamos el modelo de análisis para que encaje en nuestro medio ambiente de implementación.

Por cada objeto en el modelo de análisis, le asignamos un bloque en el modelo de diseño. En la primera parte del proceso de construcción trabajamos principalmente con los bloques.

El primer intento de un modelo de diseño se puede hacer mecánicamente, basado en el modelo de análisis. Inicialmente cada objeto del análisis llega a ser un bloque. Esta regla de transformación significa que obtenemos una traza clara en los modelos. La traza es una propiedad importante en el desarrollo de sistema.

La comunicación entre los bloques queda establecida a través de estímulos los cuales son enviados desde un bloque a otro para activar su ejecución. En el transcurso de

esta ejecución pueden enviarse otros estímulos a otros bloques. Los **diagramas de interacción** permiten describir esta secuencia de varios estímulos de bloques. El Modelo Use Case puede ser utilizado para construir estos diagramas de interacción. Para cada caso de uso, el diagrama describe en detalle cómo y cual estimulo debe ser enviado y en que orden. De esta forma el modelo de Diseño consiste en una completa descripción de los bloques con sus respectivas interfaces.

Los diagramas de interacción le dan una habilidad única al diseñador para ver la secuencia completa en un caso de uso a un nivel de apreciación global.

El primer caso lo llamamos un diagrama bifurcado. Este tipo se caracteriza por el hecho de que un objeto actúa como una araña en un tejido y así controla los otros objetos. Un diagrama bifurcado indica una estructura centralizada. Típicamente la secuencia de control se pone en un objeto, a menudo un objeto control. Otros objetos se usan típicamente como transportadores de información o como una interfaz para un usuario.

El otro caso lo llamamos un diagrama escalera. Se caracteriza este tipo por delegar responsabilidad. Cada objeto sólo conoce uno poco de los otros objetos y conoce que objetos pueden ayudar con una conducta específica. Aquí no tenemos objeto céntrico. El diagrama escalera a menudo indica una estructura descentralizada Figura 7. Cada objeto tiene una tarea separada y sólo conoce los objetos circundantes que necesita para ayudar a llevar esta tarea.

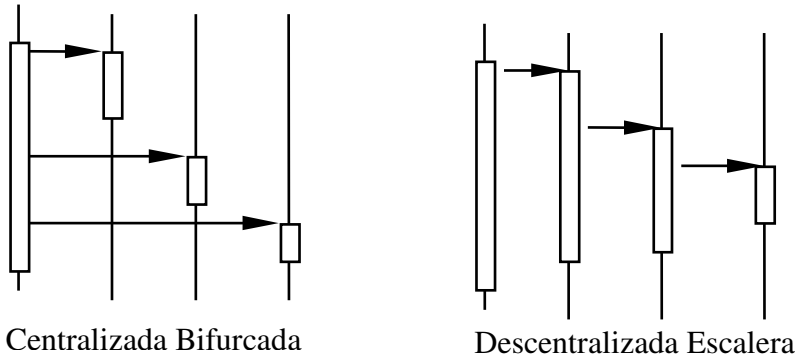


Figura 7. Estructura de diagramas de interacción.

Modelo de Implementación

El modelo de implementación consiste en el código fuente del sistema. La base de la implementación es el modelo de diseño. Aquí hemos especificado la interfaz de cada bloque y también hemos descrito la conducta qué es esperada atrás de esta interfaz. La habilidad para usar componentes es una herramienta de implementación muy poderosa.

Los objetos pueden ser implementados usando codificaciones fuente desarrolladas previamente. Llamamos tales partes componentes. Tales componentes a menudo son más simples para crear en un medio ambiente orientado a objeto, debido a la integración de funciones y datos.

El desarrollo es un proceso incremental y se deben hacer muchas iteraciones antes de que todos los estímulos se definan y se pueden inutilizar las interfaces.

Modelo de Prueba

Este es el último modelo desarrollado en el sistema, donde se describe el resultado de la evaluación del mismo. El proceso de prueba consiste en la verificación del sistema desarrollado, es decir, en determinar en qué medida el sistema responde o no al los requerimientos de los usuarios.

Las pruebas son aplicadas a distintos niveles del sistema. Inicialmente, los diseñadores realizan las Pruebas de Unidad en los niveles más bajos del sistema (ej. Clases, bloque o paquetes de servicios). Luego se realizan las Pruebas de Integración para ver si las diferentes unidades están trabajando conjuntamente de manera correcta. Por último, se aplica la Prueba del Sistema que no es más que una prueba de integración a nivel de todo el sistema. El esquema general se muestra en la Figura 8.

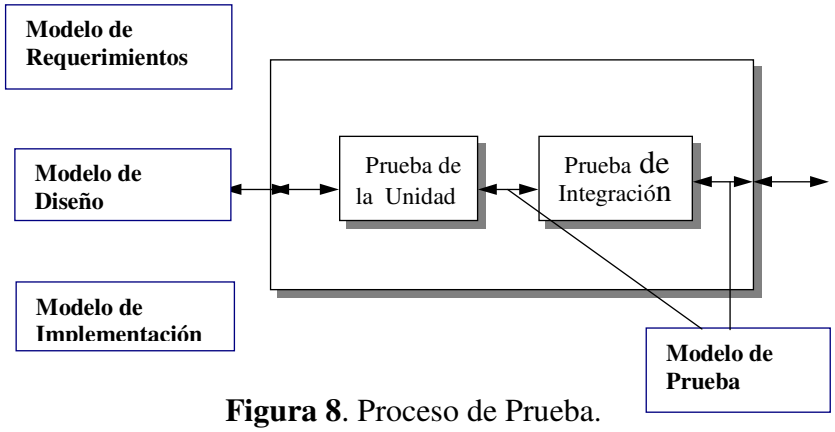


Figura 8. Proceso de Prueba.

Tipos de Prueba

Hay muchos tipos de pruebas, los cuales no son independientes la una de la otra.

- *Prueba de la unidad* significa que una y solo una unidad se prueba como tal. Esta prueba requiere que la unidad sea independiente de otras unidades. La unidad puede ser un procedimiento o una clase, pero puede ser también un módulo o un grupo de módulos. Incluimos clases, bloques y paquetes de servicio en estas unidades.
- *Prueba de integración* envuelve pruebas con el propósito de verificar que las unidades trabajen juntas correctamente. La integración y pruebas de unidad pueden ser las mismas pruebas (ejemplo, para un bloque), es el método de la prueba que difiere. Usamos casos de uso para este tipo de prueba. Los bloques, paquetes de servicio, subsistemas y el sistema completo se prueban en esta manera.
- *Una prueba de regresión* se hace cuando se han hecho cambios en el sistema, por ejemplo corrigiendo una falta, y el propósito de la prueba es verificar que la funcionalidad vieja permanece.
- *La prueba de operación* es la prueba de escala más común grande. Aquí el sistema se prueba en operación normal para un tiempo largo. Se usa el sistema en la manera intencional.

- *Una prueba de escala* llena es la continuación natural de una prueba de operación. La prueba significa que corremos el sistema en su máxima escala. Esta prueba requiere mucho del sistema, pero estos requerimientos deben ser manejados por el sistema.
- *Prueba de rendimiento o prueba de capacidad* tiene el propósito de medir la habilidad del proceso de sistema. Se diseñará la prueba para que se pueda medir el rendimiento ejecución con cargas diferentes.
- *Prueba stress* significa que se prueban los límites extremos del sistema. Una prueba de carga excesiva es un tipo especial de prueba stress y se le relata también para la prueba de rendimiento.
- *Una prueba negativa* es un tipo de prueba stress para sujetar el sistema a stress más allá de los que se han construido.
- *Pruebas basadas* en especificación de requerimiento son aquellas que se puede rastrear directamente para la especificación del requerimiento. Pueden ser unas de las pruebas anteriores.
- *Pruebas ergonómicas*: Si el sistema tiene una interfase hombre máquina entonces se deben probar los aspectos ergonómicos.
- *Prueba de la documentación del usuario* es un tipo de prueba ergonómica en que se prueba la documentación del sistema. Tanto el manual de usuario como la documentación para mantenimiento y servicio se deben probar.
- *Prueba de aceptación* es normalmente ejecutada por la organización ordenada del sistema y es el chequeo para el cliente. Ahora se prueba el sistema con datos reales. Esto es a menudo también llamada la validación del sistema. Este chequeo a menudo se hace contra la especificación del requerimiento original. A veces la especificación del requerimiento es deficiente y por consiguiente este chequeo no se debe hacer servilmente. Este tipo de prueba es a menudo también llamada prueba alfa. Se puede hacer esta prueba por un tiempo más largo cuando el sistema trabaja en el medio ambiente para el que se ha desarrollado. Si no hay ningún ordenador específico, por ejemplo un producto compilador, se usa la prueba beta. Esto significa que el producto es probado especialmente seleccionando clientes potenciales quienes usan el sistema e

informan las faltas que descubren. La prueba beta se hace antes de que el producto se lanza al mercado y es una forma de prelanzamiento.

Prueba de la Unidad

Una prueba de unidad es la forma más primitiva de prueba y es hecha normalmente por el diseñador. Para evitar equivocaciones acerca de qué es una unidad: una prueba de unidad incluye las pruebas de clases, bloques y de paquetes del servicio.

Cuando se prueba una unidad hay generalmente dos métodos:

Prueba Estructural

El propósito de la prueba estructural es probar que la estructura interior es correcta. Esto significa que tú usas tu conocimiento de cómo se lleva a cabo la unidad cuando se prueba. La prueba estructural es a veces también llamada prueba basada en programa, la prueba de caja blanca o la prueba de caja de vidrio.

Prueba de Especificación

La prueba de especificación, o la prueba de caja negra, tiene el propósito de verificar las relaciones entrada / rendimiento de una unidad. La meta es verificar la conducta específica de la unidad, qué es, qué hace la unidad, pero no nos interesamos ahora en cómo la unidad resuelve esto

.

Prueba de Integración

El propósito de la prueba de integración es probar si las unidades diferentes que se han desarrollado trabajan juntas de manera apropiada.

En la prueba de integración incluimos pruebas de casos de uso, subsistemas y el sistema completo. Se pueden incluir paquete de servicio y pruebas del bloque también hasta cierto punto, ya que también integra unidades.

Las pruebas de paquetes de servicio y bloques significan tanto las pruebas de unidad como las pruebas de integración.

La prueba de integración es una actividad que podemos describir como se muestra en la Figura 9. Uno comienza el trabajo planificando la prueba. Este plan es la base para identificar qué se probará y entonces especificarlo en más detalle. Esta especificación es la base de la ejecución real.

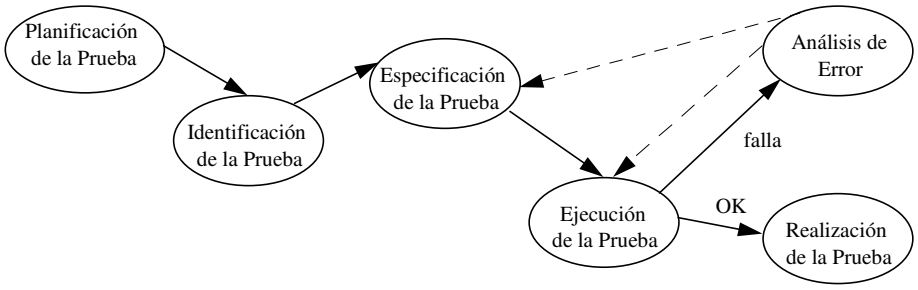


Figura 9. Actividades en la prueba de integración.

- **Planificación de la Prueba.**

La planificación puede empezar cuando comenzamos el desarrollo. La planificación de la prueba debe considerar también cualquier norma para la prueba y los recursos requeridos para cada subprueba.

Una prueba log se guarda durante el trabajo de la prueba completa. Un log se debe conectar a una versión del sistema. El propósito del log es dar una historia del estudio breve de todas las actividades de la prueba, tanto éxitos como fracasos. Los fracasos a menudo requieren un texto más largo explicativo indicando la razón por el fracaso y explicando qué acción se ha tomado. El log se archiva después de la prueba y es la base del refinamiento del proceso de prueba y la planificación de pruebas nuevas.

- **Identificación de la Prueba**.

La identificación es encontrar qué se probará. Para el primer tiempo varias clases, bloques, paquetes del servicio y subsistemas se traen juntos y por eso la prueba debe concentrarse en esto. Cada caso de uso se prueba separadamente en un principio. Cuando todos los casos de uso se han probado (a varios niveles) el sistema se prueba en su integridad.

La identificación de la prueba también significa que comenzamos esbozando la especificación de la prueba. Durante la identificación dividimos la prueba en tipos de prueba diferentes. Para un caso del uso tenemos las siguientes pruebas:

(1) Pruebas de curso básico,

(2) Pruebas de curso casual,

(3) Pruebas basado en la especificación de requerimiento,

(4) Prueba de la documentación del usuario.

Las pruebas del sistema se dividen en las siguientes pruebas:

(1) Pruebas de operación,

(2) Pruebas de la escala llena,

(3) Pruebas Negativas,

(4) Pruebas basado en la especificación del requerimiento,

(5) Prueba de la documentación del usuario.

- **Especificación de la Prueba**

La especificación de la prueba también contiene las condiciones de prueba que aplica; camas de prueba, software del sistema, hardware, equipo de prueba, y versiones de ellos. La especificación también incluye el criterio para una prueba aceptada.

La base de la especificación emana del diagrama de interacción. En ellos podemos ver todos los estímulos que se envían entre el usuario y el sistema y entre los objetos en el sistema.

Se usa la especificación de la prueba para la planificación y ejecución de las pruebas. De la especificación podemos identificar la necesidad para equipo juntos con los programas de prueba y los datos de la prueba que se debe preparar.

La especificación de la prueba contiene las condiciones para la prueba y qué pruebas se hacen en qué orden. Para cada subprueba se hace una descripción detallada de cómo la prueba se ejecutará. Especificamos también el resultado esperado y el criterio para una prueba aceptada.

- **Ejecución de la Prueba**

Se hace la prueba para ejecutar las pruebas automáticas y para hacer pruebas manuales según las direcciones. La especificación de la prueba indica el resultado esperado. Si una de las subprueba debe fallar, se interrumpe la subprueba y se nota el resultado, se analiza el defecto y se corrige si es posible. Entonces se ejecuta la subprueba de nuevo.

Por medio de una tabla de decisión podemos hacer un avalúo del resultado de la prueba. La tabla incluye todas las subpruebas, pesadas según su importancia, y podemos determinar si la prueba es aceptada o no por su resultado. La importancia denota el peso de la prueba.

Un informe de prueba contiene un resumen de la prueba y es también el informe final del caso de uso y pruebas del subsistema. Consta de un resumen y el resultado de las subpruebas individuales. Los sumarios deben ser breves y deben contener conclusiones; los recursos gastados, y si la prueba es aceptada o rechazada.

- **Análisis de Error**

Si se descubren faltas cuando se hace la prueba entonces la prueba se debe analizar y se debe identificar la razón de la falta. La falta no requiere ser debido al sistema, pero puede tener otras causas:

¿Se ha ejecutado la prueba correctamente?

¿Hay una falta en la prueba de datos o el programa prueba?

¿Es causado el fracaso por la capa de prueba?

¿Se comporta el software del sistema subyacente propiamente?

- **Realización de la Prueba**

Cuando se ha completado toda la prueba, el equipo y la capa de prueba se debe restaurar para que se puedan usar de nuevo para prueba más tarde. La documentación de la prueba es tan natural preservarla como la codificación fuente y toda otra documentación.

UML (UNIFIED MODELING LANGUAGE)

UML es una especificación de notación orientada a objetos. Se basa en las anteriores especificaciones BOOCH, RUMBAUGH y COAD-YOURDON. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos, son los que representan la arquitectura del proyecto.

Con UML nos debemos olvidar del protagonismo excesivo que se le da al diagrama de clases, este representa una parte importante del sistema, pero solo representa una vista estática, es decir, muestra al sistema parado. Sabemos su estructura, pero no sabemos que le sucede a sus diferentes partes, cuando el sistema empieza a funcionar. UML introduce nuevos diagramas que representa una visión dinámica del sistema. Es decir, gracias al diseño de la parte dinámica del sistema podemos darnos cuenta en la fase de diseño de problemas de la estructura al propagar errores o de las partes que necesitan ser sincronizadas, así como del estado de cada una de las instancias en cada momento. El diagrama de clases continua siendo muy importante, pero se debe tener en cuenta que su representación es limitada, y que ayuda a diseñar un sistema robusto con partes reutilizables, pero no a solucionar problemas de propagación de mensajes ni de sincronización o recuperación ante estados de error. En resumen, un sistema debe estar bien diseñado, pero también debe funcionar bien.

UML también intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado común para todos los

desarrollos se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.

UML es ahora un Standard, no existe otra especificación de diseño orientado a objetos, ya que es el resultado de las tres opciones existentes en el mercado. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otro ramo.

UML permite la modificación de todos sus miembros mediante estereotipos y restricciones. Un estereotipo nos permite indicar especificaciones del lenguaje al que se refiere el diagrama de UML. Una restricción identifica un comportamiento forzado de una clase o relación, es decir mediante la restricción estamos forzando el comportamiento que debe tener el objeto al que se le aplica.

Diagramas. Vistazo general:

La explicación se basará en los diagramas, en lugar de en vistas o anotación, ya que son estos la esencia de UML. Cada diagrama usa la anotación pertinente y la suma de estos diagramas crean las diferentes vistas. Las vistas existentes en UML son:

- Vista casos de uso: Se forma con los diagramas de casos de uso, colaboración, estados y actividades.
- Vista de diseño: Se forma con los diagramas de clases, objetos, colaboración, estados y actividades.
- Vista de procesos: Se forma con los diagramas de la vista de diseño. Recalcando las clases y objetos referentes a procesos.
- Vista de implementación: Se forma con los diagramas de componentes, colaboración, estados y actividades.
- Vista de despliegue: Se forma con los diagramas de despliegue, interacción, estados y actividades.

Se dispone de dos tipos diferentes de diagramas, los que dan una vista estática del sistema y los que dan una visión dinámica. Los diagramas estáticos son:

- Diagrama de clases: muestra las clases, interfaces, colaboraciones y sus relaciones. Son los más comunes y dan una vista estática del proyecto.
- Diagrama de objetos: Es un diagrama de instancias de las clases mostradas en el diagrama de clases. Muestra las instancias y como se relacionan entre ellas. Se da una visión de casos reales.
- Diagrama de componentes: Muestran la organización de los componentes del sistema. Un componente se corresponde con una o varias clases, interfaces o colaboraciones.
- Diagrama de despliegue.: Muestra los nodos y sus relaciones. Un nodo es un conjunto de componentes. Se utiliza para reducir la complejidad de los diagramas de clases y componentes de un gran sistema. Sirve como resumen e índice.
- Diagrama de casos de uso: Muestran los casos de uso, actores y sus relaciones. Muestra quien puede hacer que y las relaciones que existen entre acciones (casos de uso). Son muy importantes para modelar y organizar el comportamiento del sistema.

Lo diagramas dinámicos son:

- Diagrama de secuencia, Diagrama de colaboración: Muestran a los diferentes objetos y las relaciones que pueden tener entre ellos, los mensajes que se envían entre ellos. Son dos diagramas diferentes, que se puede pasar de uno a otro sin perdida de información, pero que nos dan puntos de vista diferentes del sistema. En resumen, cualquiera de los dos es un Diagrama de Interacción.
- Diagrama de estados: muestra los estados, eventos, transiciones y actividades de los diferentes objetos. Son útiles en sistemas que reaccionen a eventos.
- Diagrama de actividades: Es un caso especial del diagrama de estados, muestra el flujo entre los objetos. Se utilizan para modelar el funcionamiento del sistema y el flujo de control entre objetos.

Como podemos ver, el número de diagramas es muy alto, en la mayoría de los casos, excesivos, y UML permite definir solo los necesarios, ya que no todos son necesarios en todos los proyectos. En el documento se dará una breve explicación de todos, ampliándose para los más necesarios.

Diagramas recomendados

Los diagramas a representar dependerán del sistema a desarrollar, para ello se efectúan las siguientes recomendaciones dependiendo del sistema. Estas recomendaciones se deberán adaptar a las características de cada desarrollo, y seguramente será la practica lo que nos diga las cosas que echamos en falta o los diagramas que parecen ser menos necesarios.

- Aplicación monopuesto:
  - Diagrama de casos de uso.
  - Diagrama de clases.
  - Diagrama de interacción.
- Aplicación monopuesto, con entrada de eventos:
  - Añadir: Diagrama de estados.
- Aplicación cliente servidor:
  - Añadir: Diagrama de despliegue y diagrama de componentes, dependiendo de la complejidad.
- Aplicación compleja distribuida:
  - Todos.

Así tenemos, que para una aplicación sencilla debemos realizar, entre tres y seis tipos de diagramas, y para una aplicación compleja unos nueve tipos. ¿Es esto demasiado trabajo? En un principio no lo parece, ya que el tiempo dedicado a la realización de los diagramas es proporcional al tamaño del producto a realizar, no entraremos en la discusión de que el tiempo de diseño no es tiempo perdido si no ganado. Para la mayoría de los casos tendremos suficiente con tres o cuatro diagramas. Debemos pensar que UML esta pensado para el modelado tanto de pequeños sistemas como de sistemas complejos, y debemos tener en cuenta que los sistemas complejos pueden estar compuestos por millones de líneas de código y ser realizados por equipos de centenares de programadores. Así que no debemos preocuparnos, el mayor de nuestros proyectos, desde el punto de vista de UML no deja de ser un proyecto mediano llegando a pequeño.

Diagrama de casos de uso

Se emplean para visualizar el comportamiento del sistema, una parte de el o de una sola clase. De forma que se pueda conocer como responde esa parte del sistema. El diagrama de uso es muy útil para definir como debería ser el comportamiento de una parte del sistema, ya que solo especifica como deben comportarse y no como están implementadas las partes que define. Por ello es un buen sistema de documentar partes del código que deban ser reutilizables por otros desarrolladores. El diagrama también puede ser utilizado para que los expertos de dominio se comuniquen con los informáticos sin llegar a niveles de complejidad. Un caso de uso especifica un requerimiento funcional, es decir, indica: esta parte debe hacer esto, cuando pase esto.

En el diagrama nos encontramos con diferentes figuras que pueden mantener diversas relaciones entre ellas:

- Casos de uso: representado por una elipse, cada caso de uso contiene un nombre, que indique su funcionalidad. Los casos de uso pueden tener relaciones con otros casos de uso. Sus relaciones son:

  - Include: Representado por una flecha, en el diagrama de ejemplo podemos ver como un caso de uso, el de totalizar el coste incluye a dos casos de uso.

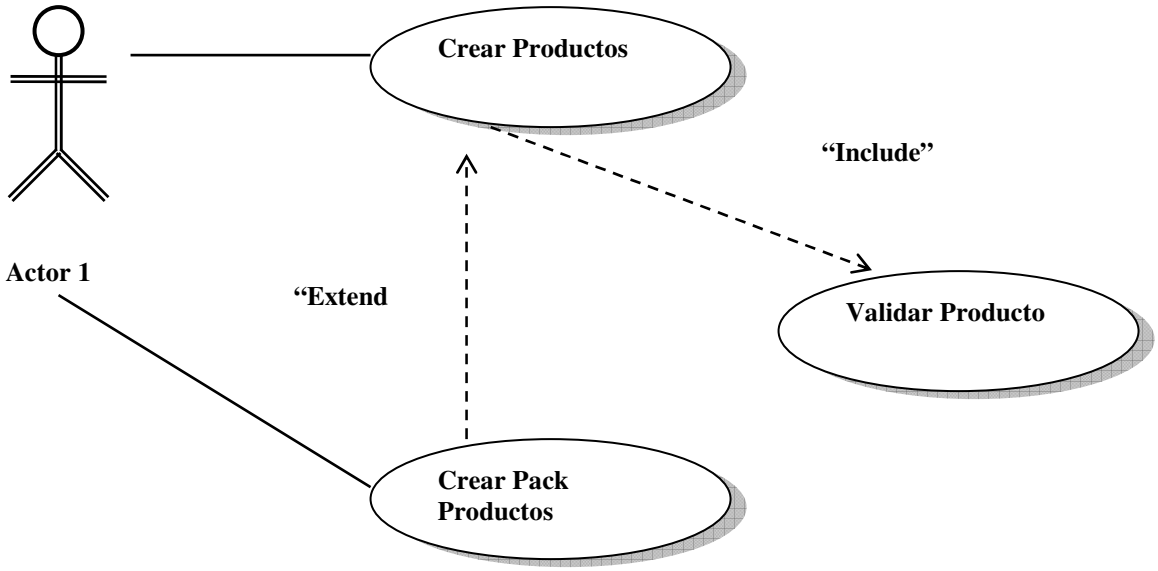
  - Extends: Una relación de una caso de Uso A hacia un caso de uso B indica que el caso de uso B implementa la funcionalidad del caso de uso A.

  - Generalization: Es la típica relación de herencia.

- Actores: se representan por un muñeco. Sus relaciones son:

  - Communicates: Comunica un actor con un caso de uso, o con otro actor.

- Parte del sistema (System boundary): Representado por un cuadro, identifica las diferentes partes del sistema y contiene los casos de uso que la forman.



En este grafico encontramos tres casos de usos: Crear producto, utiliza validar producto, y Crear pack productos es una especialización de Crear productos.

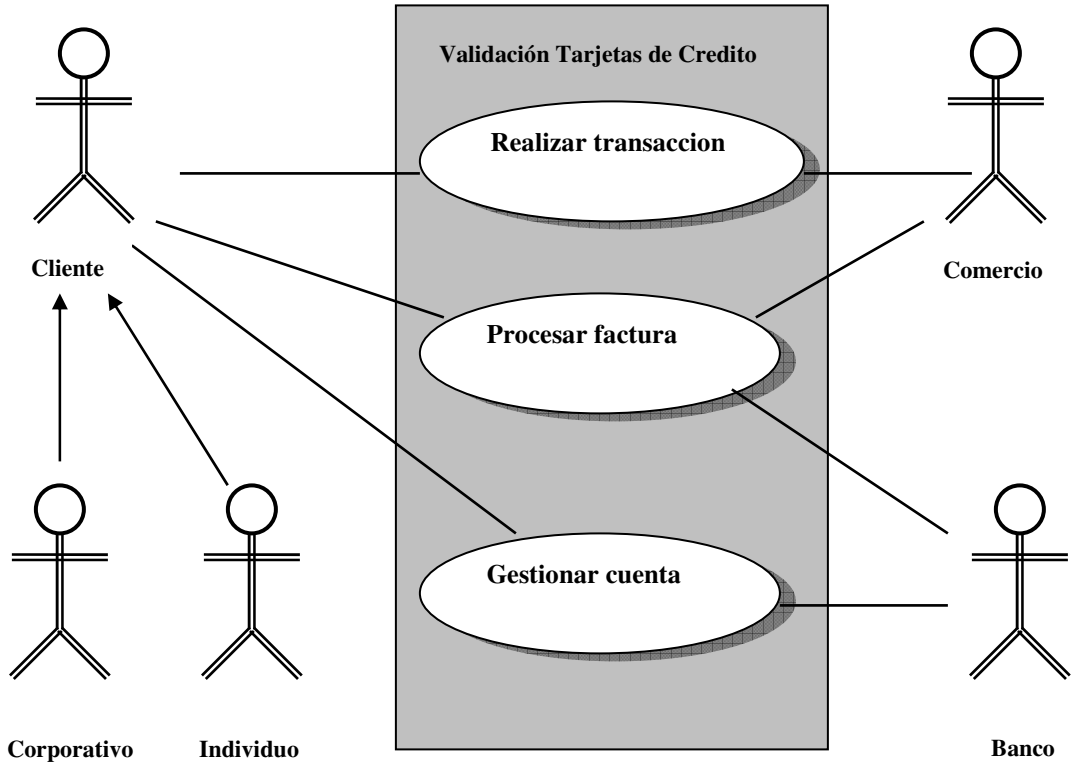
Podemos emplear el diagrama de dos formas diferentes, para modelar el contexto de un sistema, y para modelar los requisitos del sistema.

Modelado del contexto

Se debe modelar la relación del sistema con los elementos externos, ya que son estos elementos los que forman el contexto del sistema.

Los pasos a seguir son:

- Identificar los actores que interactúan con el sistema.
- Organizar a los actores.
- Especificar sus vías de comunicación con el sistema.



Modelado de requisitos

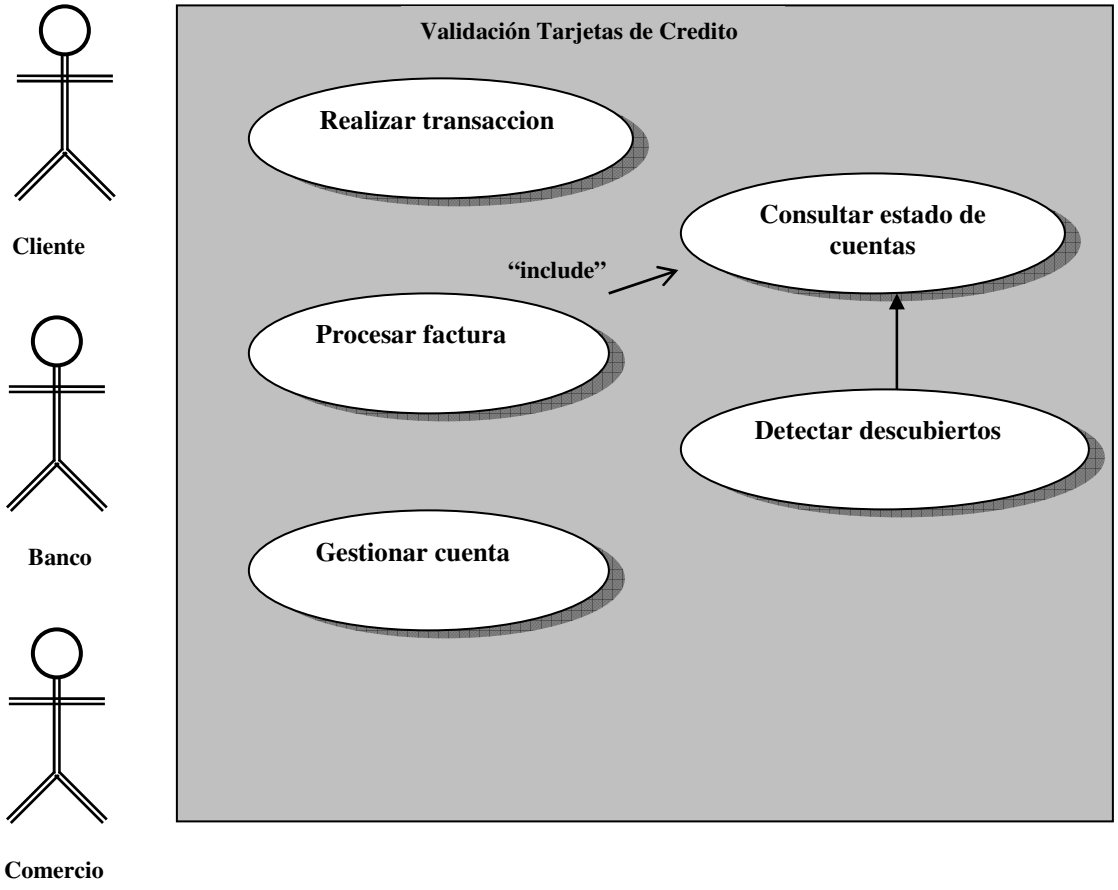
La función principal, o la mas conocida del diagrama de casos de uso, es documentar los requisitos del sistema, o de una parte de el.

Los requisitos establecen un contrato entre el sistema y su exterior, definen lo que se espera que realice el sistema, sin definir su funcionamiento interno. Es el paso siguiente al modelado del contexto, no indica relaciones entre autores, tan solo indica cuales deben ser las funcionalidades (requisitos) del sistema. Se incorporan los casos de uso necesarios que no son visibles desde los usuarios del sistema.

Para modelar los requisitos es recomendable:

- Establecer su contexto, para lo que también podemos usar un diagrama de casos de uso.
- Identificar las necesidades de los elementos del contexto (Actores).
- Nombrar esas necesidades, y darles forma de caso de uso.

- Identificar que casos de uso pueden ser especializaciones de otros, o buscar especializaciones comunes para los casos de uso ya encontrados.



Como podemos ver se incluyen nuevos casos de uso que no son visibles por ninguno de los actores del sistema, pero que son necesarios para el correcto funcionamiento.

Diagrama de clases

Forma parte de la vista estática del sistema. En el diagrama de clases como ya hemos comentado será donde definiremos las características de cada una de las clases, interfaces, colaboraciones y relaciones de dependencia y generalización. Es decir, es donde daremos rienda suelta a nuestros conocimientos de diseño orientado a objetos, definiendo las clases e implementando las ya típicas relaciones de herencia y agregación.
En el diagrama de clases debemos definir a estas y a sus relaciones.

La Clase

Una clase esta representada por un rectángulo que dispone de tres apartados, el primero para indicar el nombre, el segundo para los atributos y el tercero para los métodos. Cada clase debe tener un nombre único, que las diferencie de las otras.

Un atributo representa alguna propiedad de la clase que se encuentra en todas las instancias de la clase. Los atributos pueden representarse solo mostrando su nombre, mostrando su nombre y su tipo, e incluso su valor por defecto.

Un método u operación es la implementación de un servicio de la clase, que muestra un comportamiento común a todos los objetos. En resumen, es una función que le indica a las instancias de la clase, que hagan algo.

Para separar las grandes listas de atributos y de métodos se pueden utilizar estereotipos.

Usuario
Nombre: char Direccion: char Situación: int = 3
Entrar () Salir () Trabajar ()

Aquí vemos un ejemplo. La clase usuario contiene tres atributos: nombre que es public, dirección que es protected y situación que es private. Situación empieza con el valor 3. También dispone de tres métodos Entrar, Salir y Trabajar.

Relaciones entre clases

Existen tres relaciones diferentes entre clases, Dependencias, Generalización y Asociación. En las relaciones se habla de una clase destino y de una clase origen. La clase origen es desde la que se realiza la acción de relacionar. Es decir, desde la que parte la

flecha, la destino es la que recibe la flecha. Las relaciones se pueden modificar con estereotipos o con restricciones.

Dependencias

Es una relación de uso, es decir una clase usa a otra, que la necesita para su cometido. Se representa con una flecha discontinua va desde la clase utilizadora a la clase utilizada. Con la dependencia mostramos que un cambio en la clase utilizada puede afectar al funcionamiento de la clase utilizadora, pero no al contrario. Aunque las dependencias se pueden crear tal cual, es decir sin ningún estereotipo (palabreja que aparece al lado de la línea que representa la dependencia) UML permite dar mas significado a las dependencias, es decir concretar mas, mediante el uso de estereotipos.

- Estereotipos de relación Clase-objeto.
- Bind: La clase utilizada es una plantilla, y necesita de parámetros para ser utilizada, con Bind se indica que la clase se instancia con los parámetros pasándole datos reales para sus parámetros.
- Derive: Se utiliza al indicar relaciones entre dos atributos, indica que el valor de un atributo depende directamente del valor de otro. Es decir el atributo edad depende directamente del atributo Fecha nacimiento.
- Friend: Especifica una visibilidad especial sobre la clase relacionada. Es decir podrá ver las interioridades de la clase destino.
- InstanceOF: Indica que el objeto origen es una instancia del destino.
- Instantiate: indica que el origen crea instancias del destino.
- Powertype: indica que el destino es un contenedor de objetos del origen, o de sus hijos.
- Refine: se utiliza para indicar que una clase es la misma que otra, pero mas refinada, es decir dos vistas de la misma clase, la destino con mayor detalle.

Generalización

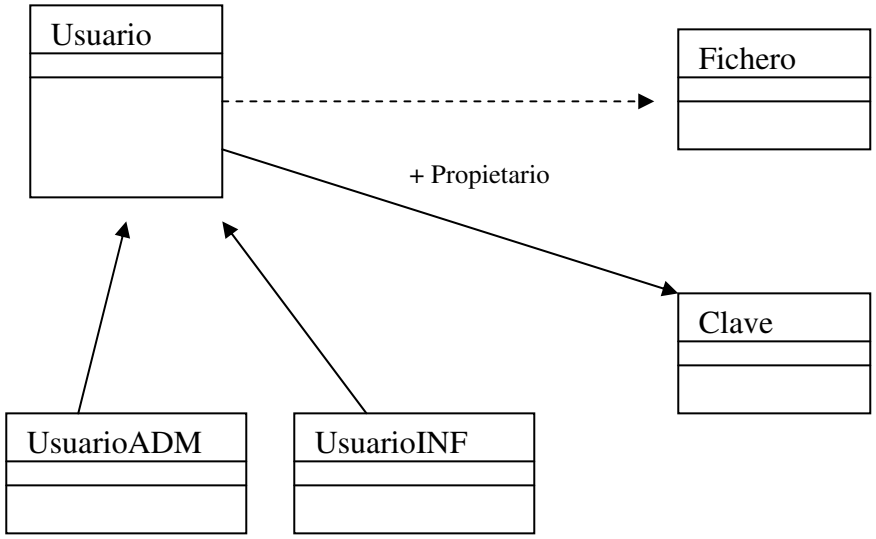
Pues es la herencia, donde tenemos una o varias clases padre o superclase o madre, y una clase hija o subclase. UML soporta tanto herencia simple como herencia múltiple.

Aunque la representación común es suficiente en el 99.73% de los casos UML nos permite modificar la relación de Generalización con un estereotipo y dos restricciones.

- Estereotipo de generalización.
- Implementation: El hijo hereda la implementación del padre, sin publicar ni soportar sus interfaces.
- Restricciones de generalización.
- Complete: La generalización ya no permite más hijos.
- Incomplete: Podemos incorporar más hijos a la generalización.
- Disjoint: solo puede tener un tipo en tiempo de ejecución, una instancia del padre solo podrá ser de un tipo de hijo.
- Overlapping: puede cambiar de tipo durante su vida, una instancia del padre puede ir cambiando de tipo entre los de sus hijos.

Asociación

Especifica que los objetos de una clase están relacionados con los elementos de otra clase. Se representa mediante una línea continua, que une las dos clases. Podemos indicar el nombre, multiplicidad en los extremos, su rol, y agregación.



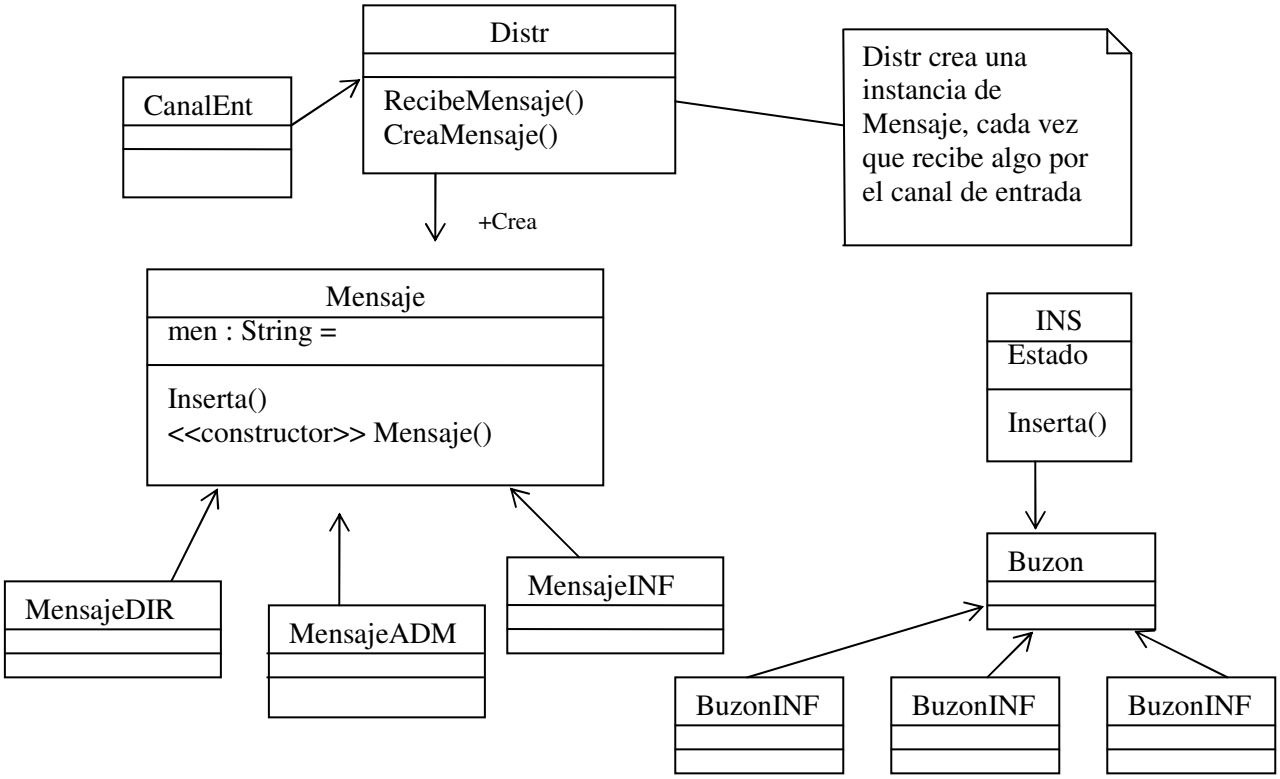
Ejemplo

En este diagrama se han creado cuatro clases. La clase principal es Usuario, que tiene dos clases hijas UsuarioADM y UsuarioINF. El usuario mantiene una relación de asociación con la clase Clave, se indica que es propietario de una clave, o de un número indeterminado de ellas. Se le crea también una relación de dependencia con la clase Perfil, es decir las instancias de usuario contendrán como miembro una instancia de Perfil.

Diagrama de objetos

Forma parte de la vista estática del sistema. En este diagrama se modelan las instancias de las clases del diagrama de clases. Muestra a los objetos y sus relaciones, pero en un momento concreto del sistema. Estos diagramas contienen objetos y enlaces. En los diagramas de objetos también se pueden incorporar clases, para mostrar la clase de la que es un objeto representado.

En este diagrama se muestra un estado del diagrama de eventos. Para realizar el diagrama de objetos primero se debe decidir que situación queremos representar del sistema. Es decir si disponemos de un sistema de mensajería, deberemos decidir que representaremos el sistema con dos mensajes entrantes, los dos para diferentes departamentos, dejando un departamento inactivo. Para el siguiente diagrama de clases:



Tendríamos un diagrama de objetos con dos instancias de Mensaje, mas concretamente con una instancia de MensajeDIR y otra de MensajeADM, con todos sus atributos valorados. También tendríamos una instancia de cada una de las otras clases que deban tener instancia. Como CanalEnt, INS, Distr, y el Buzón correspondiente a la instancia de mensaje que se este instanciando. En la instancia de la clase INS se deberá mostrar en su miembro Estado, que esta ocupado realizando una inserción.

En un diseño no podemos encontrar con multitud de diagramas de objetos, cada uno de ellos representando diferentes estados del sistema.

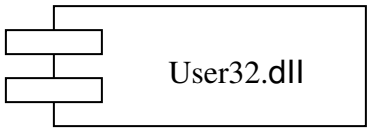
Diagrama de componentes

Se utilizan para modelar la vista estática de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama

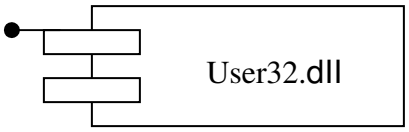
incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema.

En el situaremos librerías, tablas archivos, ejecutables y documentos que formen parte del sistema.

Uno de los usos principales es que puede servir para ver que componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.



Aquí tenemos un componente del sistema de Windows. En el diagrama de componentes de Windows debe salir este componente, ya que sin el, el sistema no funcionaría.



En esta otra figura tenemos el mismo componente, pero indicamos que dispone de una interfase. Al ser una Dll el interfase nos da acceso a su contenido. Esto nos hace pensar que la representación anterior es incorrecta, pero no es así solo corresponde a un nivel diferente de detalle.

Como ya hemos indicado antes todo objeto UML puede ser modificado mediante estereotipos, el standard que define UML son:

- Executable
- Library
- Table
- File
- Document.

Aunque por suerte no estamos limitados a estas especificaciones. Que pasa si queremos modelar un proyecto de Internet donde nuestros componentes son ASP, HTML, y Scripts, y queremos marcarlo en el modelo. Pues utilizamos un estereotipo. Existen ya unos definidos WAE (Web Applications Extensión).

Podemos modelar diferentes partes de nuestro sistema, y modelar diferentes entidades que no tiene nada que ver entre ellas.

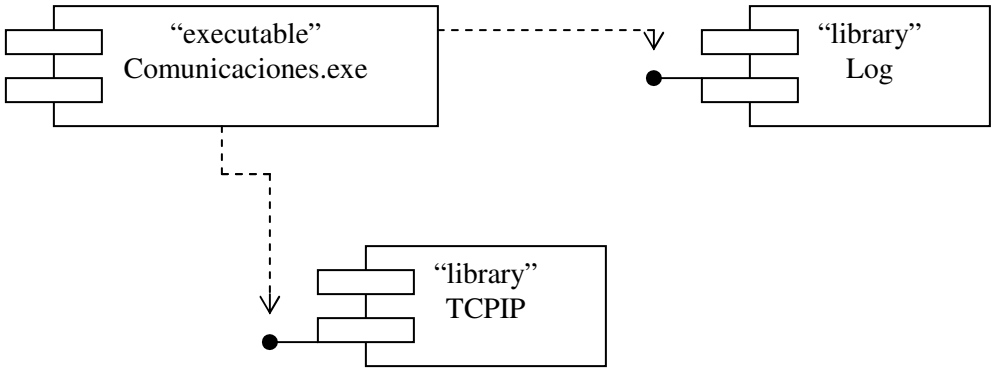
- Ejecutables y bibliotecas.
- Tablas.
- API
- Código fuente.
- Hojas HTML.

Ejecutables

Nos facilita la distribución de ejecutables a los clientes. Documenta sus necesidades y dependencias. Si disponemos de un ejecutable que solo se necesita a el mismo para funcionar no necesitaremos el diagrama de componentes.

Los pasos a seguir para modelar, a priori no a posteriori, son:

- Identificar los componentes, las particiones del sistema, cuales son factibles de ser reutilizadas. Agruparlos por nodos y realizar un diagrama por cada nodo que se quiera modelar.
- Identificar cada componente con su estereotipo correspondiente.
- Considerar las relaciones entre componentes.

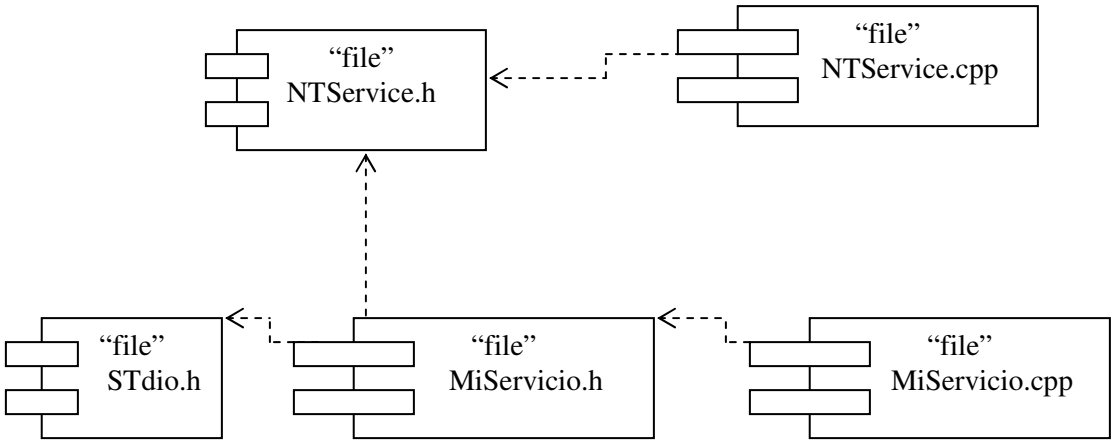


En este grafico se muestra un ejecutable que utiliza dos librerías, estas dos librerías disponen de su interfase con el que ofrecen el acceso a sus servicios. Se puede ver que estas librerías son componentes que pueden ser reutilizados en otras partes del sistema.

Código fuente

Se utiliza para documentar las dependencias de los diferentes ficheros de código fuente. Un ejecutable, o librería es una combinación de estos ficheros, y al mostrar la dependencia entre ellos obtenemos una visión de las partes necesarias para la creación del ejecutable o librería.

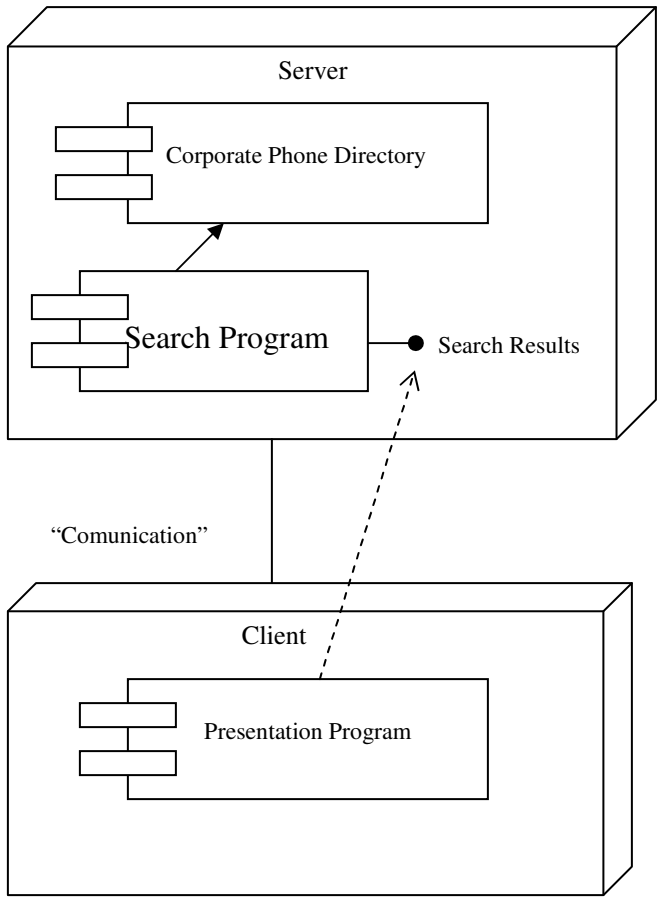
Al tener documentadas las relaciones se pueden realizar cambios en el código de un archivo teniendo en cuenta donde se utiliza, y que otros ficheros pueden verse afectados por su modificación.



Aquí tenemos la relación entre los diferentes ficheros de un sistema. Cada fichero Cpp utiliza su fichero .h correspondiente, y MiServicio.h utiliza NTService.h u Stdio.h.

Diagramas de despliegue

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Es decir se sitúa el software en el hardware que lo contiene. Cada Hardware se representa como un nodo.



Un nodo se representa como un cubo, un nodo es un elemento donde se ejecutan los componentes, representan el despliegue físico de estos componentes.

Aquí tenemos dos nodos, el cliente y el servidor, cada uno de ellos contiene componentes. El componente del cliente utiliza una interfase de uno de los componentes

del servidor. Se muestra la relación existente entre los dos Nodos. Esta relación podríamos asociarle un estereotipo para indicar que tipo de conexión disponemos entre el cliente y el servidor, así como modificar su cardinalidad, para indicar que soportamos diversos clientes.

Como los componentes pueden residir en mas de un nodo podemos situar el componente de forma independiente, sin que pertenezca a ningún nodo, y relacionarlo con los nodos en los que se sitúa.

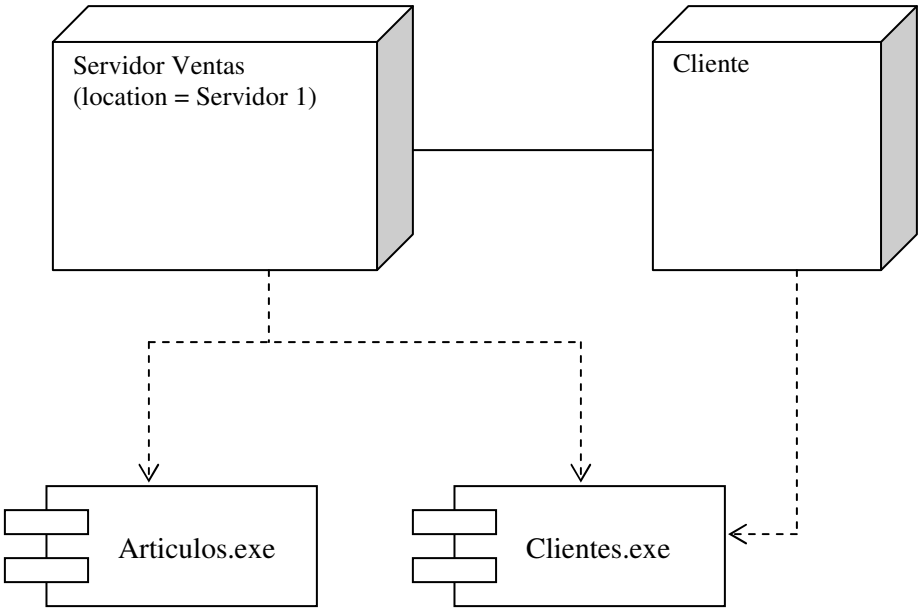


Diagrama Secuencia

El diagrama de secuencia forma parte del modelado dinámico del sistema. Se modelan las llamadas entre clases desde un punto concreto del sistema. Es útil para observar la vida de los objetos en sistema, identificar llamadas a realizar o posibles errores del modelado estático, que imposibiliten el flujo de información o de llamadas entre los componentes del sistema.

En el diagrama de secuencia se muestra el orden de las llamadas en el sistema. Se utiliza un diagrama para cada llamada a representar. Es imposible representar en un solo diagrama de secuencia todas las secuencias posibles del sistema, por ello se escoge un punto de partida. El diagrama se forma con los objetos que forman parte de la secuencia, estos se sitúan en la parte superior de la pantalla, normalmente en la izquierda se sitúa al que inicia la acción. De estos objetos sale una línea que indica su vida en el sistema. Esta línea simple se convierte en una línea gruesa cuando representa que el objeto tiene el foco del sistema, es decir cuando el esta activo.

CAPÍTULO III

SOLUCIÓN AL PROBLEMA PROPUESTO

Una vez realizado el proceso de factibilidad y escogencia de la aplicación para Palm Pilot, se debe realizar todo el proceso de similitud de las Bases de Datos implicadas y las relaciones entre las tablas de almacenamiento de datos, así como las posibles tablas de catálogos que se tengan que crear para establecer la relación entre los catálogos Palm y Sicredito, para luego poder realizar todo el proceso de programación de la interface señalada.

El sistema programado para Palm Pilot (que la organización BanGente ha decidido llamar PortaCredit) está programado bajo una plataforma de programación llamada Satellite Form 3.1, la cual posee un archivo Active de extensión .ocx que permite la comunicación entre cualquier plataforma de programación para PC y la PalmPilot al momento de la sincronización **[PALM99]**, únicamente con la activación e invocación de las funciones de este archivo en las ventanas que se estén programando. En nuestro caso estaremos trabajando con PowerBuilder. Algo importante que debemos tener en cuenta es que este archivo Active únicamente bajaría la información de la Base de Datos del sistema programado para Palm Pilot, en formato .dbf y la Base de Datos de la organización es una plataforma SQL Server.

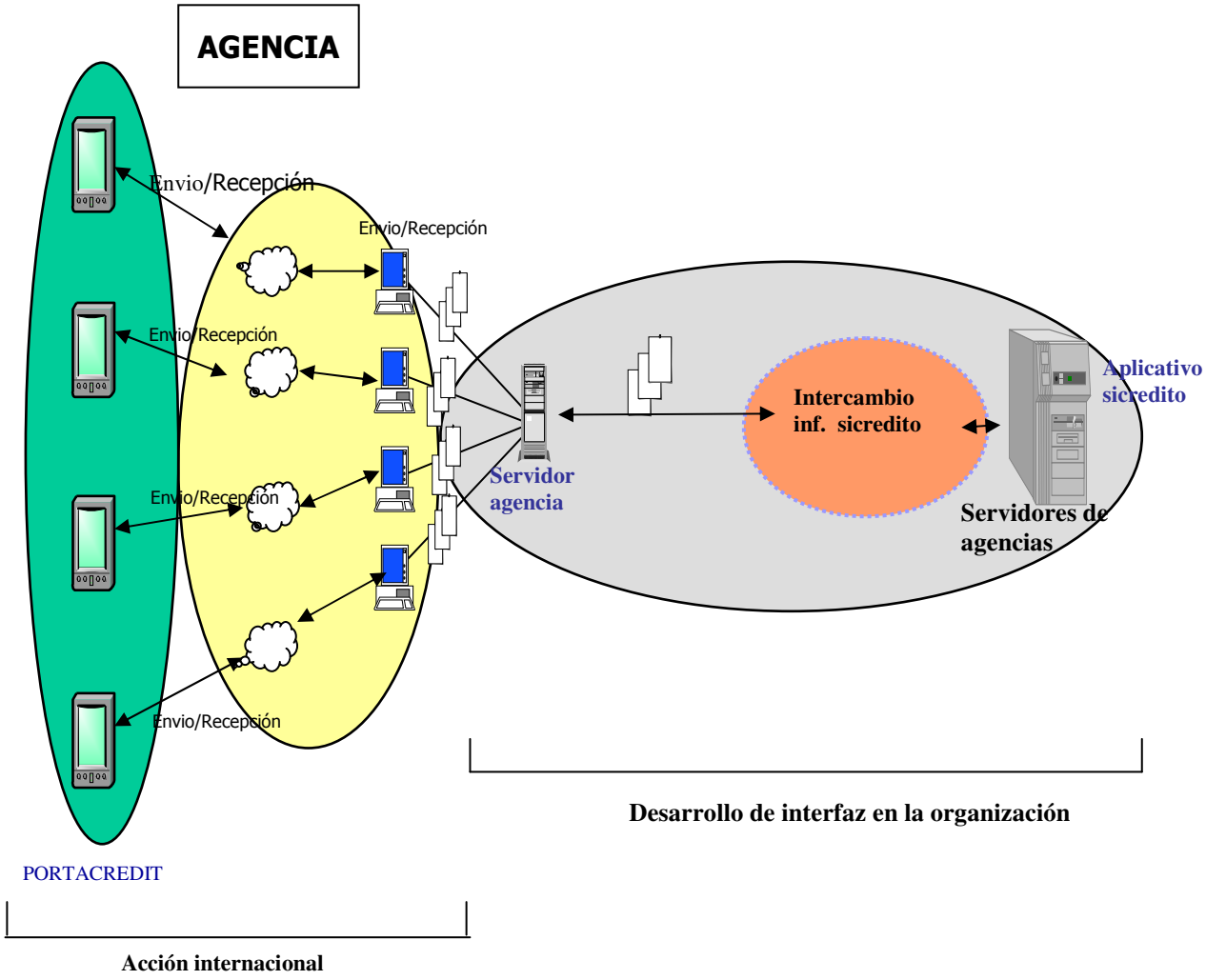


Figura 10. Arquitectura de la solución

En la Figura 10, podemos visualizar de manera gráfica como será la Arquitectura de la solución presentada. Se observa, como el envío y recepción de información se realiza entre los dispositivos Palm y los PC, luego con ayuda de la interfase, dicha información será interpretada y actualizada en las BD de SQL Server de la aplicación Sicredito.

Solución

Estableceremos en el sistema de interfaz una comunicación directa con los archivos de la BD del PortaCredit por medio de una transferencia de los datos de la Palm hacia el PC, con la diferencia que no se alimentará directamente dicha Base de Datos, sino que se creará una copia de la Base de Datos del Portacredit en la Base de Datos del sistema de Sicredito en el SMDB SQL Server , la cual puede proporcionar una manera más segura de anticiparse a los posibles errores del proceso de sincronización sin dañar los datos del asesor ubicada en su Palm, permitiendo así revisar los errores de los datos en los archivos no modificados en el PC.

Debido al volumen de la información el proceso de transformación de los datos será realizado por medio de Store Procedure para poder procesarlos lo mas rápido y eficientemente posible, aunque para que pueda ser entendible por el SQL Server se deben levantar los datos dinámicamente en una ventana o reporte de Powerbuilder para luego ser almacenada en una Base de Datos temporal creada en el SMBD SQL Server.

Para realizar la validación con el sistema central de riesgo, se establecerá un motor de búsqueda entre la Base de Datos de la Central de Riesgo y la Base de Datos del Sicredito, creando tablas especificas para ello en la BD de Sicredito, que luego alimentará la BD del Portacredit en los campos que se establezcan para ello. En la Figura 11 se puede ver gráficamente como se manejaría este proceso.

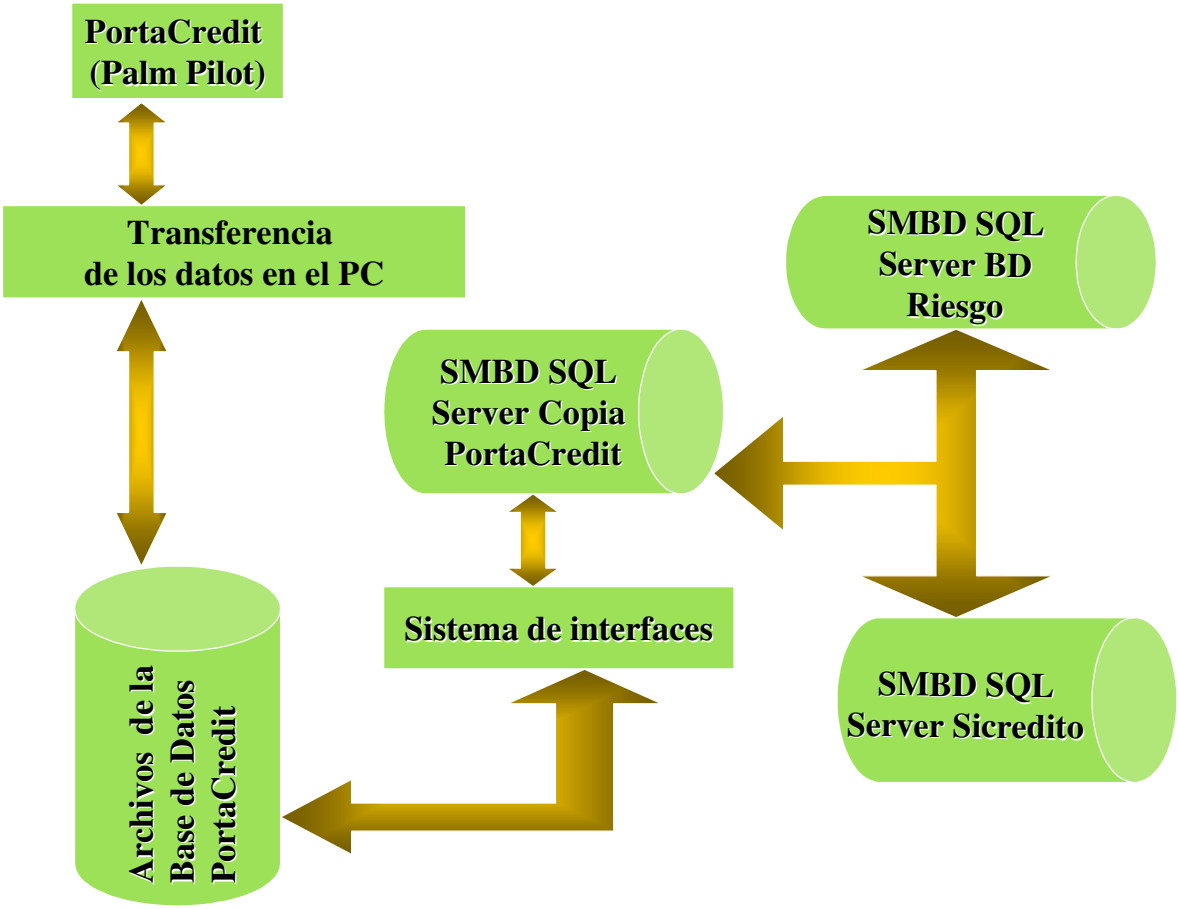


Figura 11. Ejemplo gráfico de la solución

Un punto resaltante en la investigación es que debemos establecer mecanismos de respaldo de los archivos de los asesores que contienen la información del Portacredit, a ser modificados en sus Palm´s (dada la importancia de la información que estos manejan). Preferiblemente, se debe respaldar la información antes de las modificaciones debido a cualquier problema que se pueda presentar (esta debe ser respaldada preferiblemente en el Servidor de la agencia donde se esté realizando la sincronización o hotsync, para tratar de tener respaldos físicos de dicha data, ya sea en cintas, CD's, etc.).

Cabe destacar que en reuniones con la Gerencia de Tecnología y el Coordinador de Proyectos de TI, se acordó que las plataformas a utilizar serían:

- Plataformas de programación, Power Builder 9.0
- Sistema Manejador de Base de Datos SQL Server 2005
- Sistema operativo de los PC, Windows XP

Para lograr llevar a buen fin cualquiera la solución planteada, antes de comenzar a implantarlas, se deben realizar ciertas actividades que podremos definir dentro de un plan de trabajo primordial (siguiendo las metodologías estudiadas para la Determinación de Requerimientos) y dentro de las cuales estaríamos definiendo los siguientes pasos:

1. Se deben realizar una serie de Entrevistas y/o Cuestionarios, donde se podrá tener un conocimiento amplio de los mecanismos actuales del levantamiento de información y la disponibilidad de los usuarios para el cambio tecnológico, lo cual nos ayudaría a Planificar la transferencia tecnológica de los usuarios, los departamentos técnicamente involucrados y las agencias.
2. Utilizaremos la técnica de Observación para comprender de manera eficiente los estados y mecanismos de la trascripción de datos, junto con los procesos de captación y pre-aprobación de clientes, utilizado por los Asesores de Negocios.
3. Por medio de la Revisión de Registros podremos establecer las similitudes entre los Campos de ambas aplicaciones para el desarrollo de la Interfase y las referencias entre las tablas de Catálogos de ambos sistemas.

MODELO DE REQUERIMIENTOS

En el modelo de requerimientos se definen las funcionalidades que el sistema debe ofrecer, así como su comportamiento y delimitaciones.

Modelo de casos de uso

El modelo de casos de uso describe las funcionalidades que el sistema debe ofrecer desde la perspectiva del usuario, en términos de casos de uso (use-case). Cada caso de uso identifica una funcionalidad del sistema a desarrollar y la conexión de todos los casos de uso, especifica todas las formas existentes de utilización del sistema. Los casos de uso son iniciados por actores, que representan los distintos roles que pueden desempeñar los usuarios. Cabe destacar, la diferencia que existe entre un actor y un usuario, el usuario es la persona que usa el sistema, mientras que el actor representa un cierto rol que un usuario puede desempeñar. Por lo tanto, se puede considerar un actor como una clase y los usuarios como instancias de esa clase. Además, un usuario puede representar varios roles, es decir, se desempeña como muchos actores.

Estos modelos de casos de uso de la interfase de sincronización fueron el resultado del estudio de las entrevistas realizadas a los usuarios gerenciales y operativos de la organización, con la cual se pudo diseñar eficientemente el proceso general y de interfases que se debe seguir en este proyecto.

Para el sistema programado para Palm Pilot (PortaCredit), se identifican los siguientes actores:

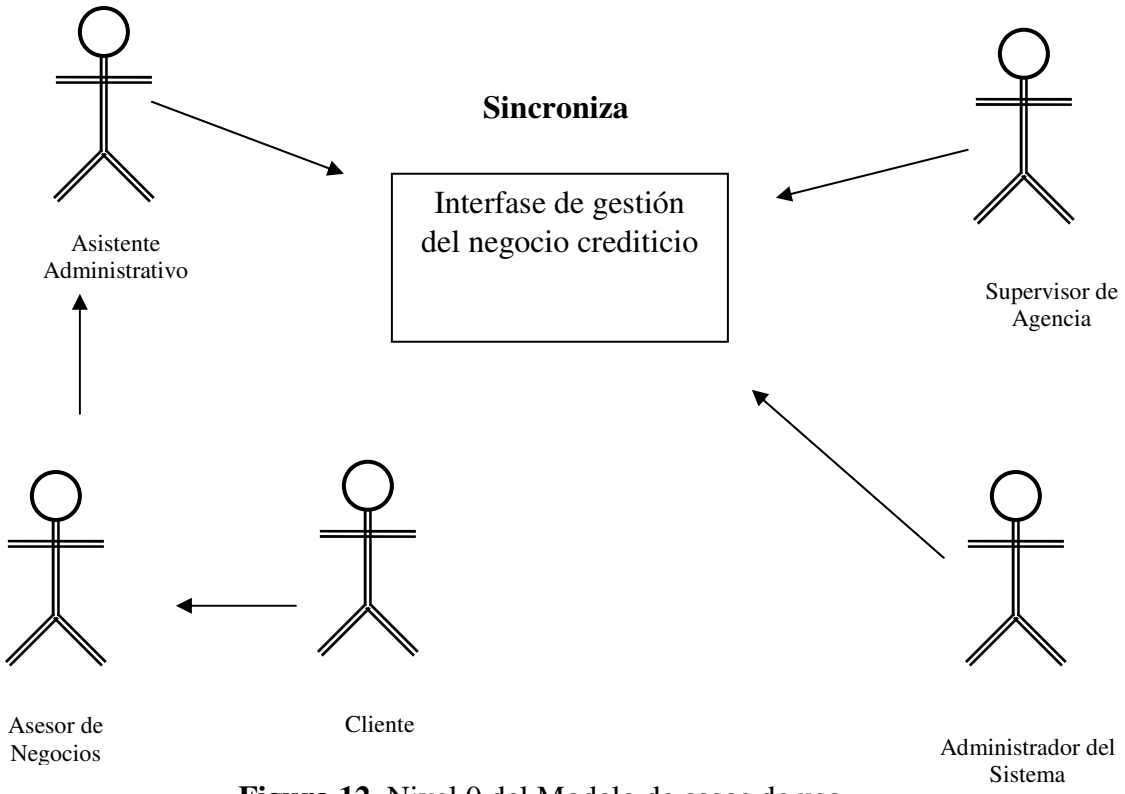


Figura 12. Nivel 0 del Modelo de casos de uso

Cliente: son todas las personas naturales que se dediquen a una o más actividades; comerciales, de servicios o de producción, ubicadas dentro del sector de la microempresa.

Asesor de Negocios: se encarga de promocionar y asesorar de manera directa a los clientes potenciales de la micro y pequeña empresa, en todos los productos activos y pasivos que ofrece BanGente, mediante una atención personalizada que permita la captación de negocios que incrementen la cartera de clientes y la prestación de un servicio óptimo, de acuerdo a las normas y procedimientos establecidos por BanGente para el cumplimiento de los objetivos planteados en la Misión y Visión de BanGente.

Asistente Administrativo: se encarga de efectuar la transcripción de los datos de los clientes al sistema de información de crédito de la sucursal.

Supervisor de Agencia: es la persona encargada de planificar, coordinar y controlar la ejecución de todas las funciones administrativas, financieras y operativas que garanticen la difusión y propagación de los programas establecidos por la institución para la prestación de óptimos servicios en la sucursal a su cargo, con el fin de lograr el cumplimiento de las metas y objetivos trazados por BanGente. Debe evaluar los créditos y solicitar su corrección para garantizar la calidad de los mismos, antes de ser discutidos en el comité.

Administrador del Sistema: es la persona encargada de atender las solicitudes de servicio que generan los usuarios. Realiza el mantenimiento de la base de datos, provee las facilidades para efectuar modificaciones de toda la información referente al Sincroniza.

Los casos de uso principales, que se han identificado para el sistema, son los siguientes:

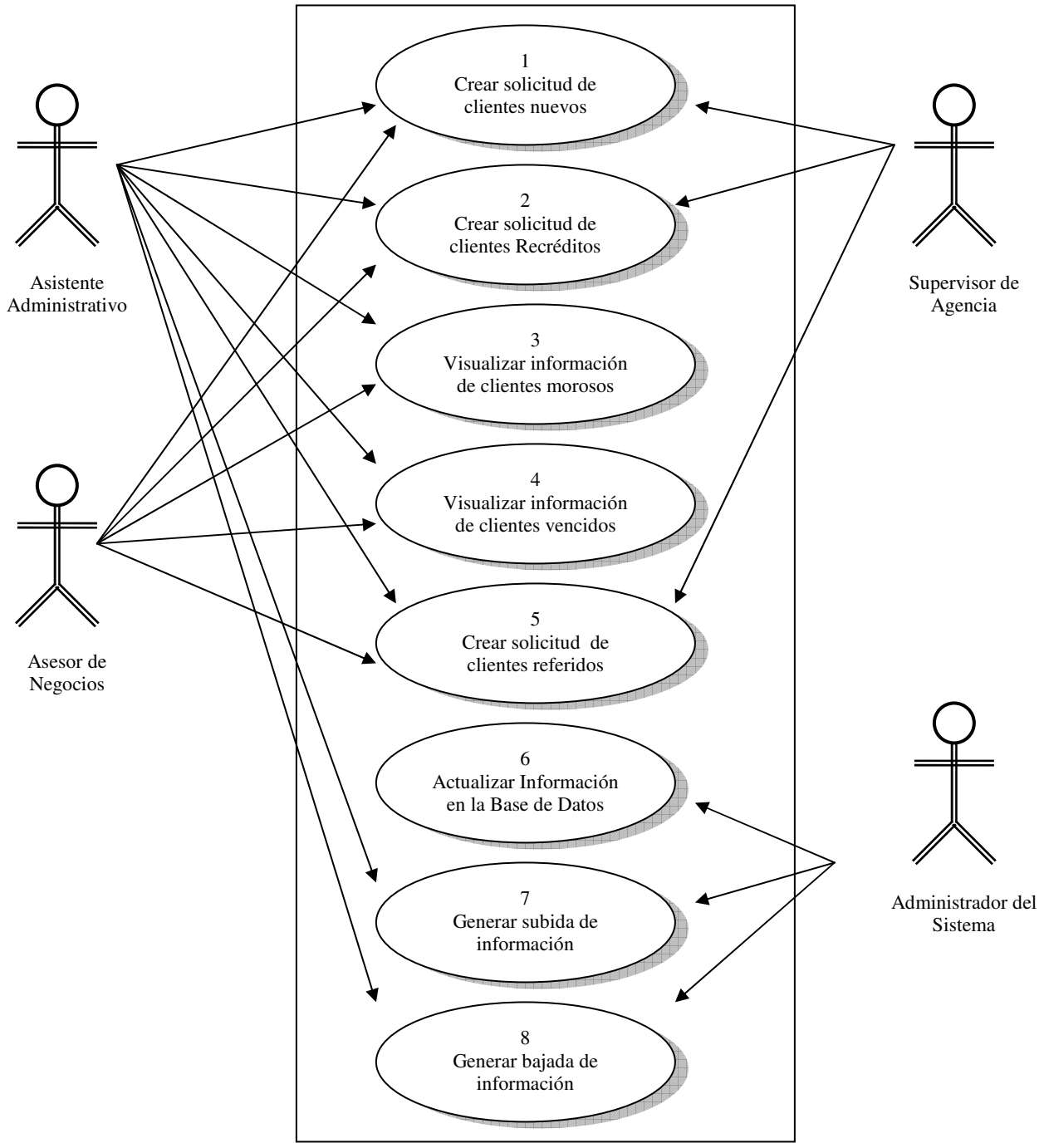


Figura 13. Nivel 1 del modelo de casos de uso

- **Crear Solicitud de Clientes Nuevos:** es el proceso de ingresar la información general de un cliente, que tiene crédito por primera vez en la Institución. *Actores Involucrados*: Asistente administrativo, Asesor de Negocios y Supervisor de Agencia.
- **Crear Solicitud de Clientes Recréditos:** es el proceso de ingresar la información general de un cliente, que ha mantenido crédito, al menos una vez, en la Institución. *Actores Involucrados*: Asistente administrativo, Asesor de Negocios y Supervisor de Agencia.
- **Visualizar Información de Clientes Morosos:** es el proceso de ver y validar la información general de un cliente, que presenta retraso en el pago del crédito. *Actores Involucrados*: Asistente administrativo y Asesor de Negocios.
- **Visualizar Información de Clientes Vencidos:** es el proceso de ver y validar la información general de un cliente, que va a renovar su crédito (posible Recrédito). *Actores Involucrados*: Asistente administrativo y Asesor de Negocios.
- **Crear Solicitud de Clientes Referidos:** es el proceso de ingresar la información general de un cliente, que es referido por otro cliente. *Actores Involucrados*: Asistente administrativo, Asesor de Negocios y Supervisor de Agencia.
- **Actualizar Información de la Base de Datos:** esta funcionalidad del sistema, permite al Administrador del Sistema, realizar el mantenimiento de la base de datos; proveer las facilidades al usuario para la consulta, ingreso, modificación, y eliminación de toda la información referente al Sistema Sincroniza. *Actores Involucrados*: Administrador del Sistema.
- **Generar Subida de Información:** es el proceso donde la Asistente Administrativa, puede efectuar la solicitud al Administrador del Sistema, de subir al servidor, la información para su modificación. *Actores Involucrados*: Asistente administrativo y Administrador del Sistema
- **Generar Bajada de Información:** es el proceso donde la Asistente Administrativa, puede efectuar la solicitud al Administrador del Sistema, de bajar al PortaCredit, la información, una vez realizada la modificación. *Actores Involucrados*: Asistente administrativo y Administrador del Sistema.

Refinamiento de los principales casos de uso:

Los casos de uso principales, descritos anteriormente, pueden refinarse en casos de uso más simples, cuya funcionalidad es más específica.

A continuación, se presenta el refinamiento de los principales casos de uso, identificados para la aplicación:

Crear Solicitud de Clientes Nuevos:

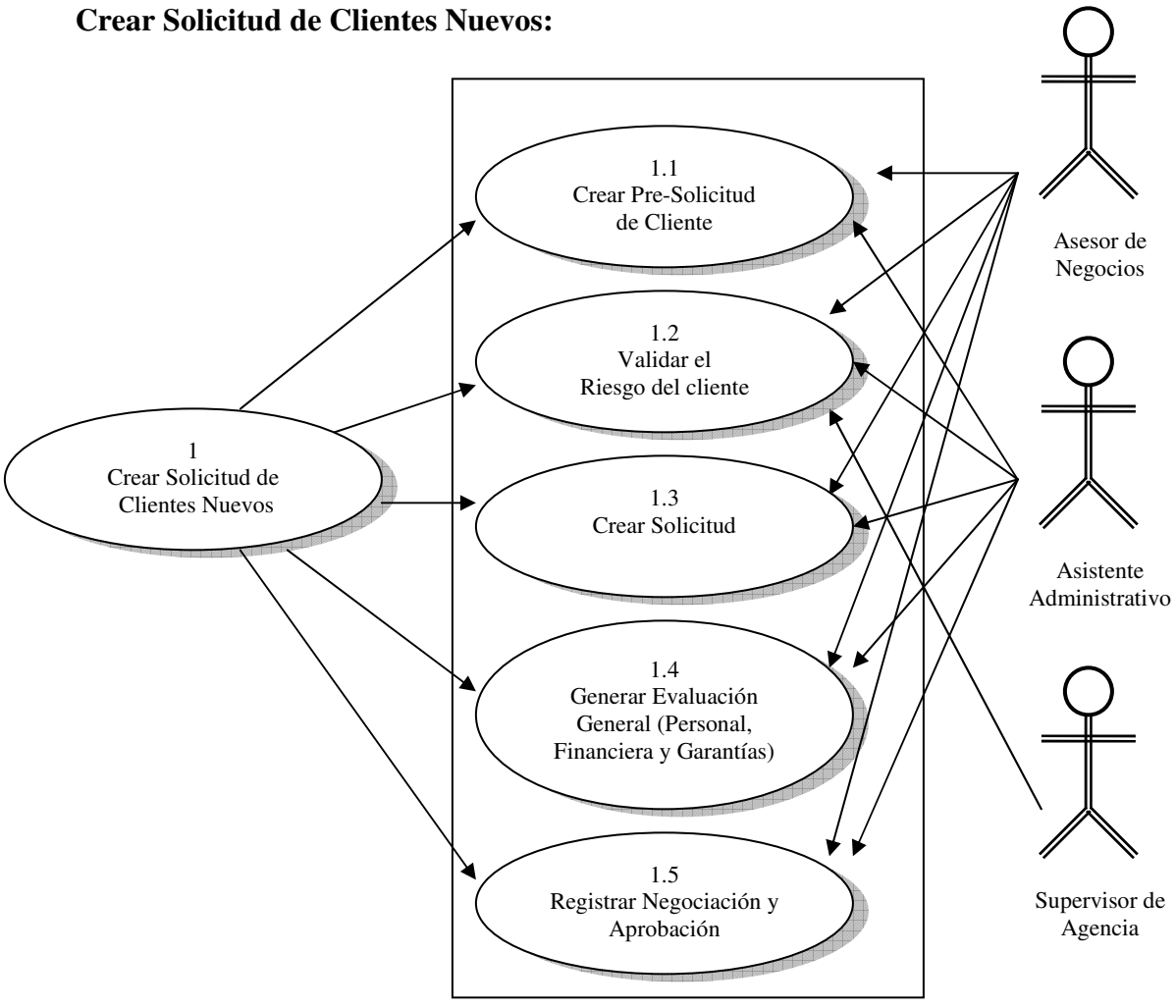


Figura 14. Nivel 2. Caso de uso Crear Solicitud Clientes Nuevos

- **Crear Pre-Solicitud de Cliente:** permite ingresar la información general de un cliente, la cual se clasifica como Pre-solicitud, permitiendo filtrar clientes que no

cumplan con las políticas de crédito de la Institución y la validación de la Central de Riesgos. *Actores Involucrados*: Asistente administrativo y Asesor de Negocios.

- **Validar el Riesgo del Cliente:** es el proceso donde se verifican los datos del cliente en la Central de Riesgo de la organización. *Actores Involucrados*: Asistente administrativo, Asesor de Negocios y Supervisor de Agencia.
- **Crear Solicitud:** permite ingresar la información general de un cliente, que cumple con los requisitos de la pre-solicitud y la validación de dicho cliente en la central de riesgo de la organización. *Actores Involucrados*: Asistente administrativo y Asesor de Negocios.
- **Generar Evaluación General:** este proceso permite el ingreso de la información de la evaluación general del cliente, comenzando con la evaluación financiera (estados financieros), la evaluación crediticia, la evaluación personal (familiar) y terminando con la evaluación de las garantías. *Actores Involucrados*: Asistente administrativo y Asesor de Negocios.
- **Registrar Negociación y Aprobación:** en esta fase se presenta un resumen de la información financiera del cliente y permite al Asesor de Negocios negociar con el cliente, mediante una función de simulación del crédito a otorgar. Adicionalmente, permite la aprobación de la solicitud de crédito, una vez que el Asesor de Negocios, obtiene la conformidad del cliente, sobre las condiciones del crédito en cuestión, cerrando así el proceso de creación de dicha solicitud. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.

Crear Solicitud de Clientes Recréditos

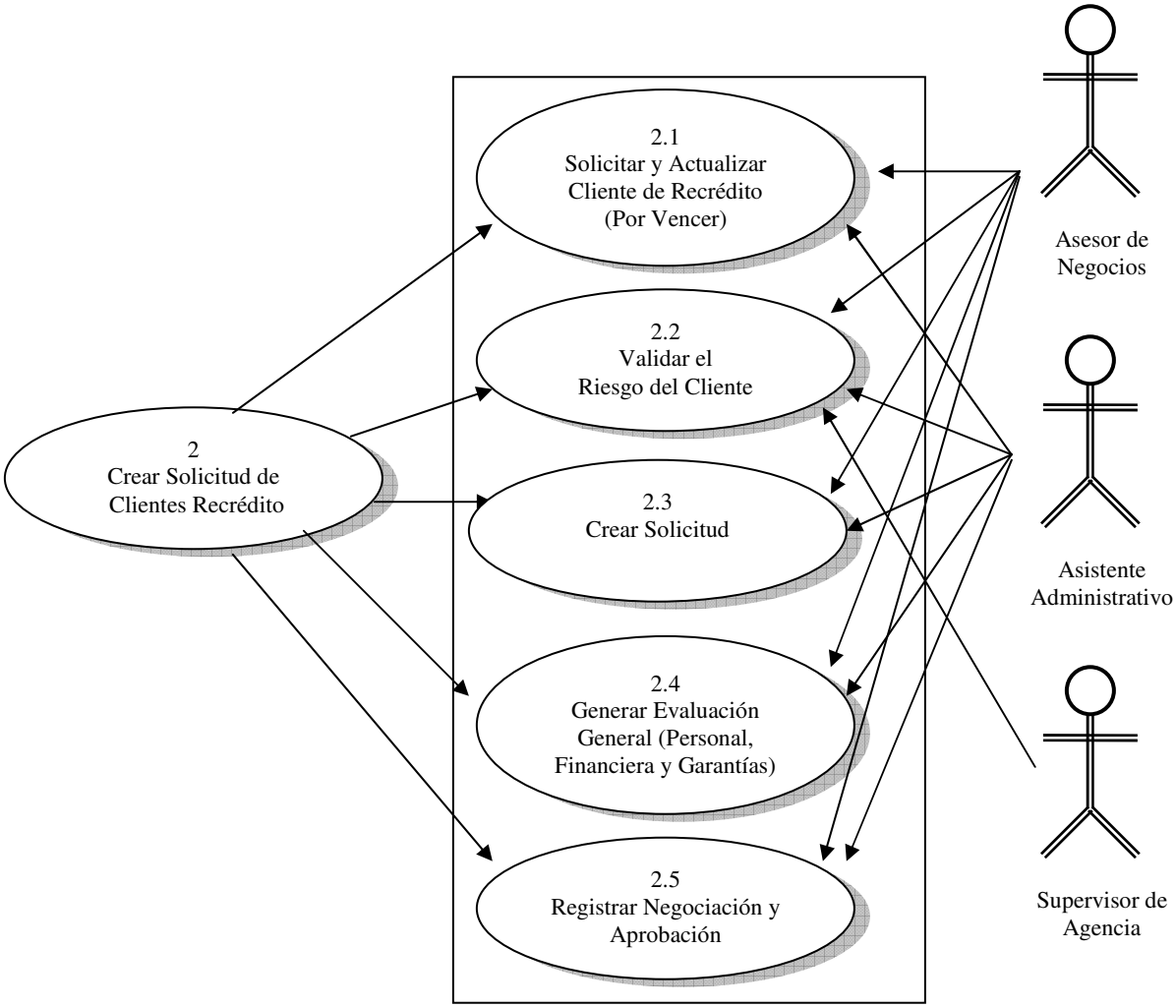


Figura 15. Nivel 2. Caso de uso Crear Solicitud Clientes Recrédito

- **Solicitar y Actualizar Cliente Recrédito (Por Vencer):** permite actualizar la información general de un cliente que va a renovar su crédito. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.
- **Validar el Riesgo del cliente:** es el proceso donde se verifican los datos del cliente en la Central de Riesgo de la organización. *Actores Involucrados*: Asesor de Negocios, Supervisor de Agencia y Asistente administrativo.
- **Actualizar Solicitud:** permite actualizar y modificar la información general de un cliente, la validación del mismo en la Central de Riesgo de la organización. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.

- **Generar Evaluación General:** este proceso permite el ingreso de la información de la evaluación general del cliente, comenzando con la evaluación financiera (estados financieros), la evaluación crediticia, la evaluación personal (familiar) y terminando con la evaluación de las garantías. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.
- **Registrar Negociación y Aprobación:** en esta fase se presenta un resumen de la información financiera del cliente y permite al Asesor de Negocios negociar con el cliente, mediante una función de simulación del crédito a otorgar.

  Adicionalmente, permite la aprobación de la solicitud de crédito, una vez que el Asesor de Negocios obtiene, la conformidad del cliente, sobre las condiciones del crédito en cuestión , cerrando así el proceso de creación de dicha solicitud. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.

Visualizar Información de Clientes Morosos

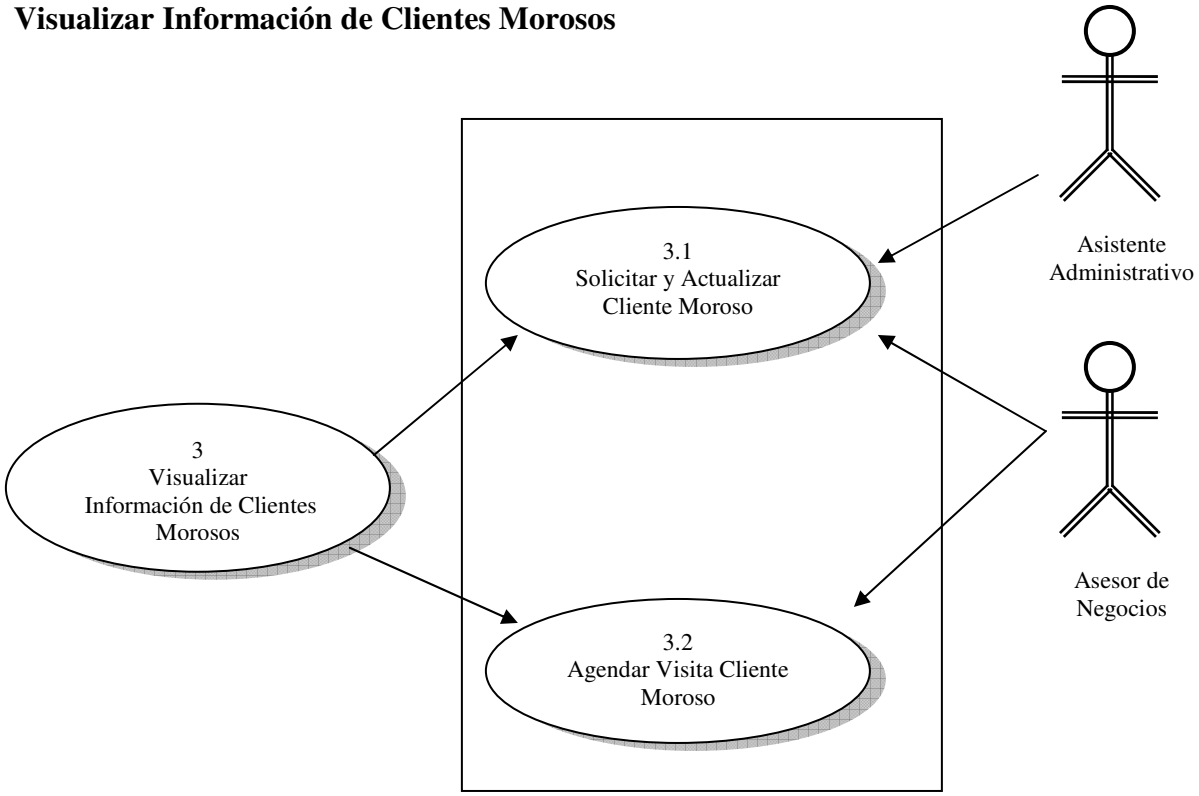


Figura 16. Nivel 2. Caso de uso Visualizar Información Clientes Morosos

- **Solicitar y Actualizar Cliente Moroso:** se despliega una lista de todos los clientes morosos, aquellos clientes que presentan retraso en el pago del crédito; permite

consultar los datos del crédito moroso. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.

- **Agendar Visita al Cliente Moroso:** permite programar una visita al cliente. *Actores Involucrados*: Asesor de Negocios.

Visualizar Información de Clientes Vencidos

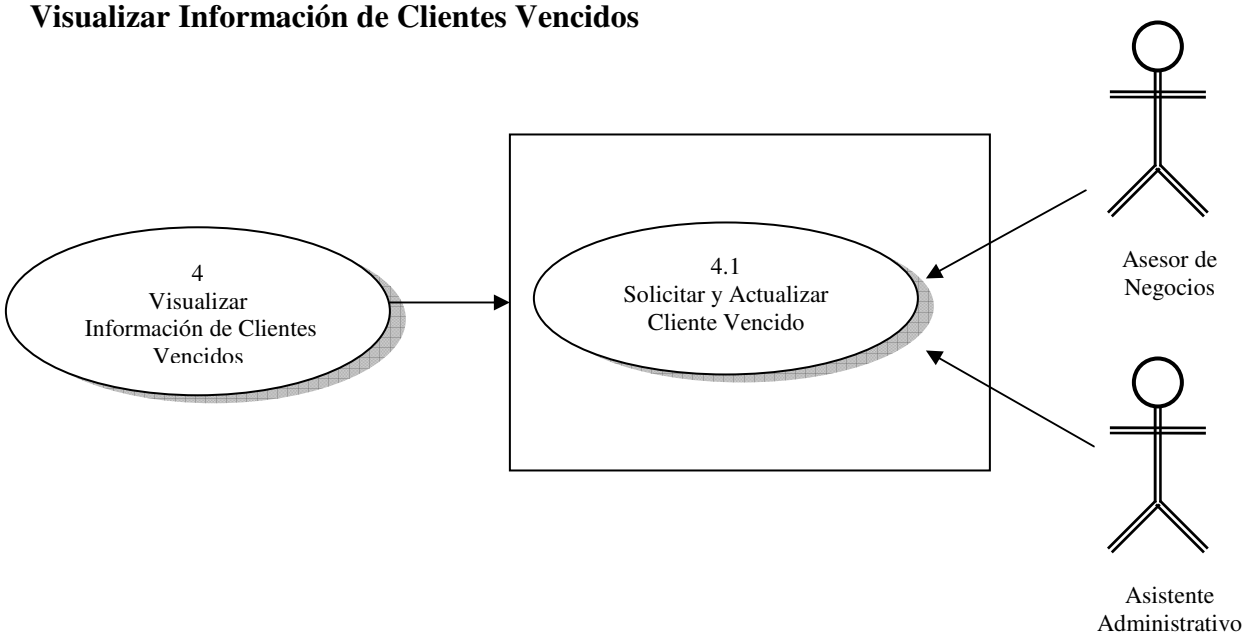


Figura 17. Nivel 2. Caso de uso Visualizar Información Clientes Vencidos

- **Solicitar y Actualizar Cliente Vencido:** se despliega una lista de los clientes que van a renovar sus créditos; permite consultar los datos del crédito y programar una visita al cliente. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.

Crear Solicitud de Clientes Referidos

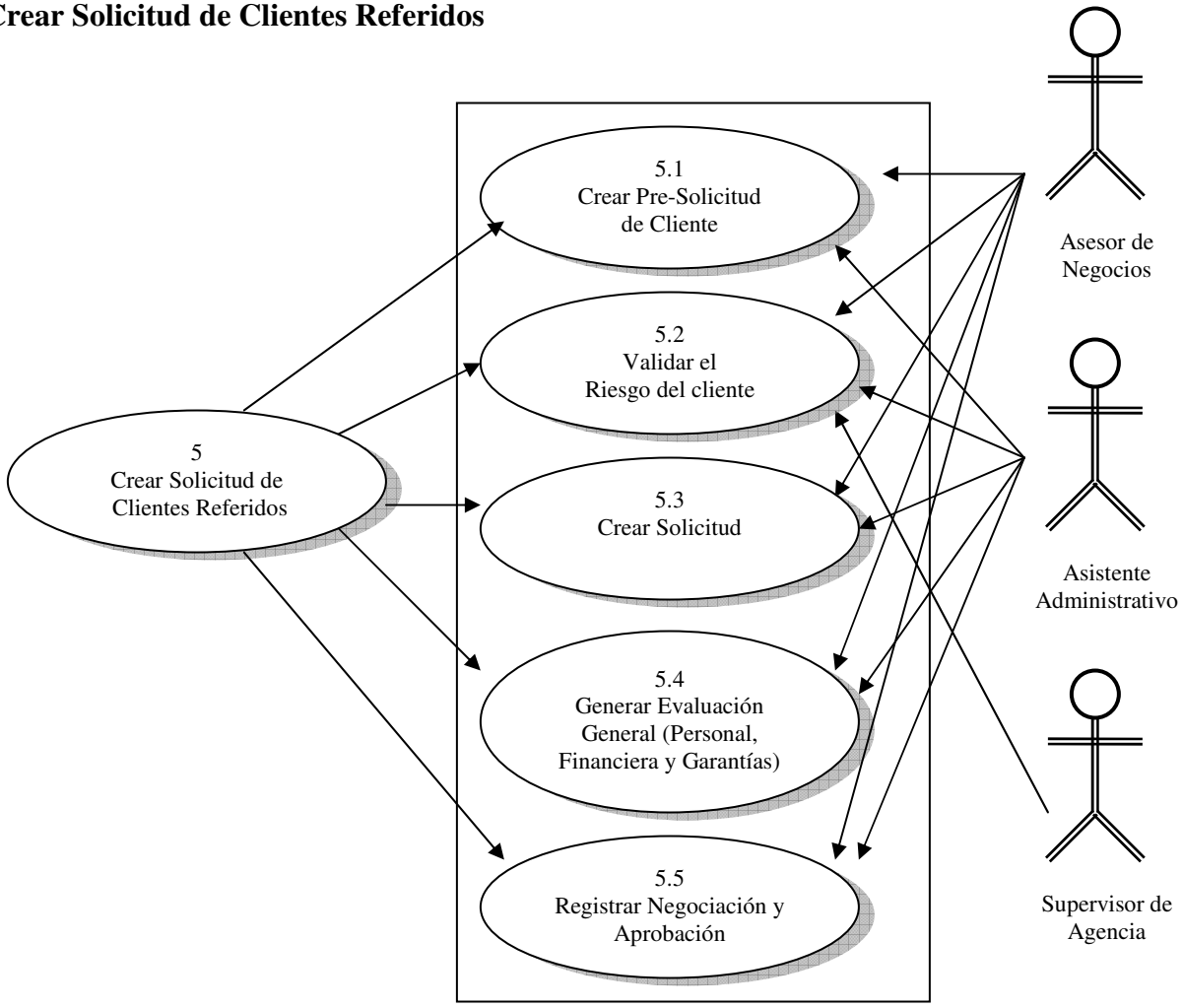


Figura 18. Nivel 2. Caso de uso Crear Solicitud Clientes Referidos.

- **Crear Pre-Solicitud de Cliente:** permite ingresar la información general de un cliente, la cual se clasifica como Pre-solicitud, permitiendo filtrar clientes que no cumplan con las políticas de crédito de la Institución y la validación de la Central de Riesgos. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.
- **Validar el Riesgo del Cliente:** es el proceso donde se verifican los datos del cliente en la Central de Riesgo de la organización. *Actores Involucrados*: Asesor de Negocios, Supervisor de Agencia y Asistente administrativo.
- **Crear Solicitud:** permite ingresar la información general de un cliente, que cumple con los requisitos de la pre-solicitud y la validación de la Central de Riesgo. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.

- **Generar Evaluación General:** este proceso permite el ingreso de la información de la evaluación general del cliente, comenzando con la evaluación financiera (estados financieros), la evaluación crediticia, la evaluación personal (familiar) y terminando con la evaluación de las garantías. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.
- **Registrar Negociación y Aprobación:** en esta fase se presenta un resumen de la información financiera del cliente y permite al Asesor de Negocios negociar con el cliente, mediante una función de simulación del crédito a otorgar. Adicionalmente, permite la aprobación de la solicitud de crédito, una vez que el Asesor de Negocios, obtiene la conformidad del cliente, sobre las condiciones del crédito en cuestión, cerrando así el proceso de creación de dicha solicitud. *Actores Involucrados*: Asesor de Negocios y Asistente administrativo.

Generar Subida de Información

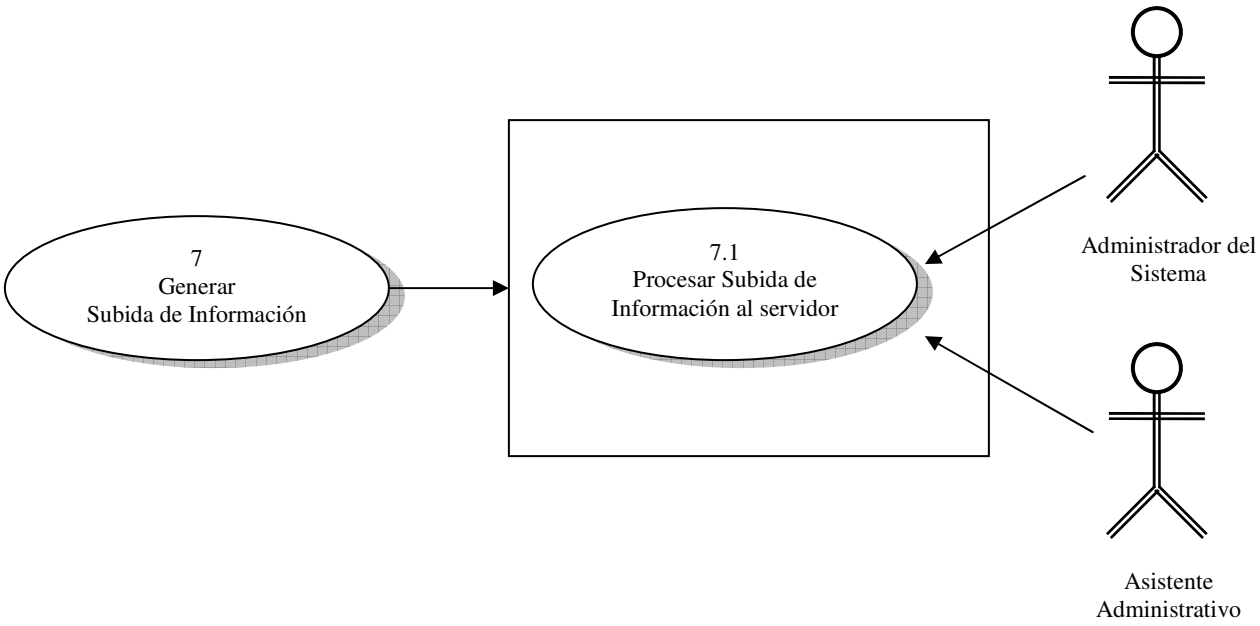


Figura 19. Nivel 2. Caso de uso Generar Subida de Información

- **Procesar Subida de Información:** permite a la Asistente Administrativo, procesar en conjunto con el Administrador del Sistema, la subida de la información al servidor de la data del Portacredit, para que este pueda realizar las modificaciones en la información general del cliente en caso de algún problema con dicha información. *Actores Involucrados*: Administrador del Sistema y Asistente administrativo.

Generar Bajada de Información

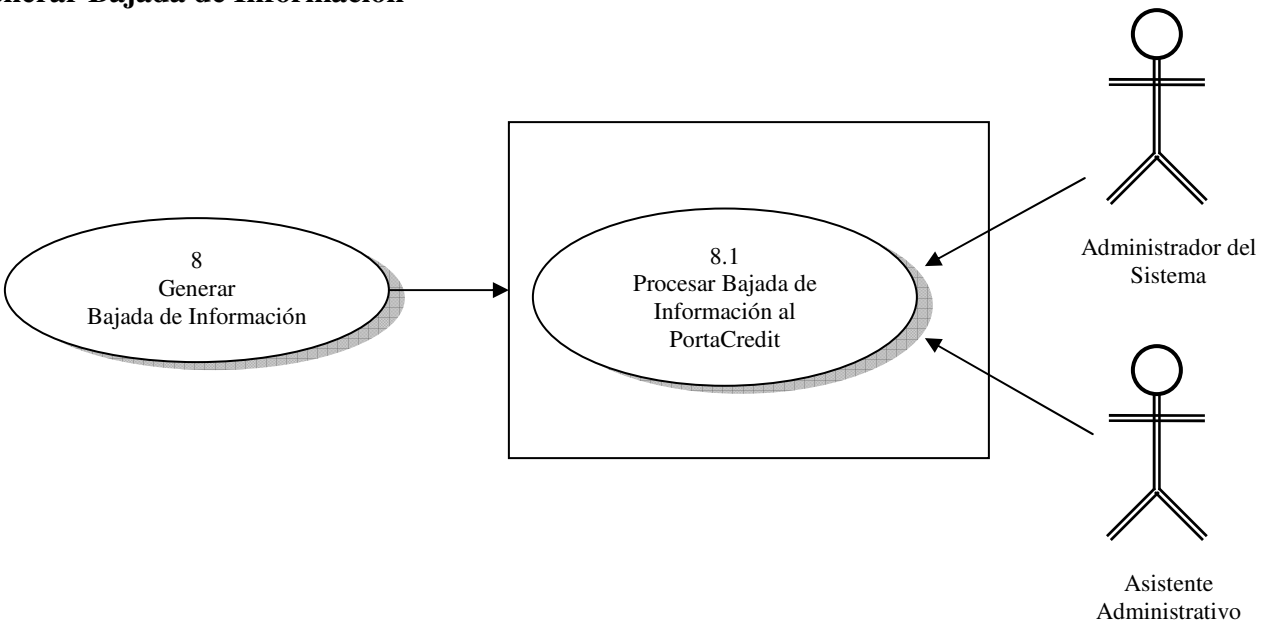


Figura 20. Nivel 2. Caso de uso Crear Generar Bajada de Información

- **Procesar Bajada de Información:** permite a la Asistente Administrativo, procesar en conjunto con el Administrador del Sistema, la bajada de la información al PortaCredit, una vez que dicho administrador haya realizado las modificaciones en la información general del cliente en caso de algún problema con dicha información. *Actores Involucrados*: Administrador del Sistema y Asistente administrativo.

Proceso de Levantamiento de información y colocación de solicitudes de crédito

Mediante las entrevistas con los usuarios y la técnica de Observación, se pudo obtener información de los procesos actuales y generar todo el proceso deseable para el manejo de la interfase de sincronización que van a utilizar los asesores de negocio.

En este proceso se quiere simular el efecto de la interfase de sincronización desarrollada y el cambio que va generar a los procesos actuales de manejo de información en la plataforma Sicredito, en el cual se elimina el manejo de papel por parte del asesor de negocios y el proceso de trascripción por parte del personal de Operaciones de oficina, en este proceso se establecen dos sincronizaciones por cada asesor de negocio de la oficina, una al momento del ingreso de información de las presolicitudes de crédito por parte del asesor de negocio en el sistema Portacredit el cual debe dirigirse a su oficina para que el personal de operaciones (Asistentes Administrativas) sincronice la información y se hagan los primeros registros de información del cliente en conjunto con la validación en la central de riesgo, lo cual le da seguridad al asesor de que el cliente es valido para optar por un crédito en la institución; y una segunda sincronización, donde, luego que el asesor de negocios a introducido toda la información crediticia (información socio económica del cliente) en el sistema Portacredit, el personal de operaciones realiza la sincronización y la interfase se encarga de complementar la información faltante en el sistema Sicredito de dichos clientes, generando así las solicitudes de crédito que serán validadas por las asistentes administrativas y generadas para prosigan con su proceso normal de aprobación y documentación de crédito.

En la figura 21 se detalla gráficamente dicho proceso expresado anteriormente en conjunto con el proceso de aprobación y documentación de crédito, igualmente se visualiza el proceso de cobro de cuotas que posteriormente alimentara a través de la interfase los reportes de morosos, vencidos y renovaciones del sistema Portacredit que le estarán sirviendo de insumo al asesor de negocio para realizar efectivamente su gestión y aumentar así su productividad.

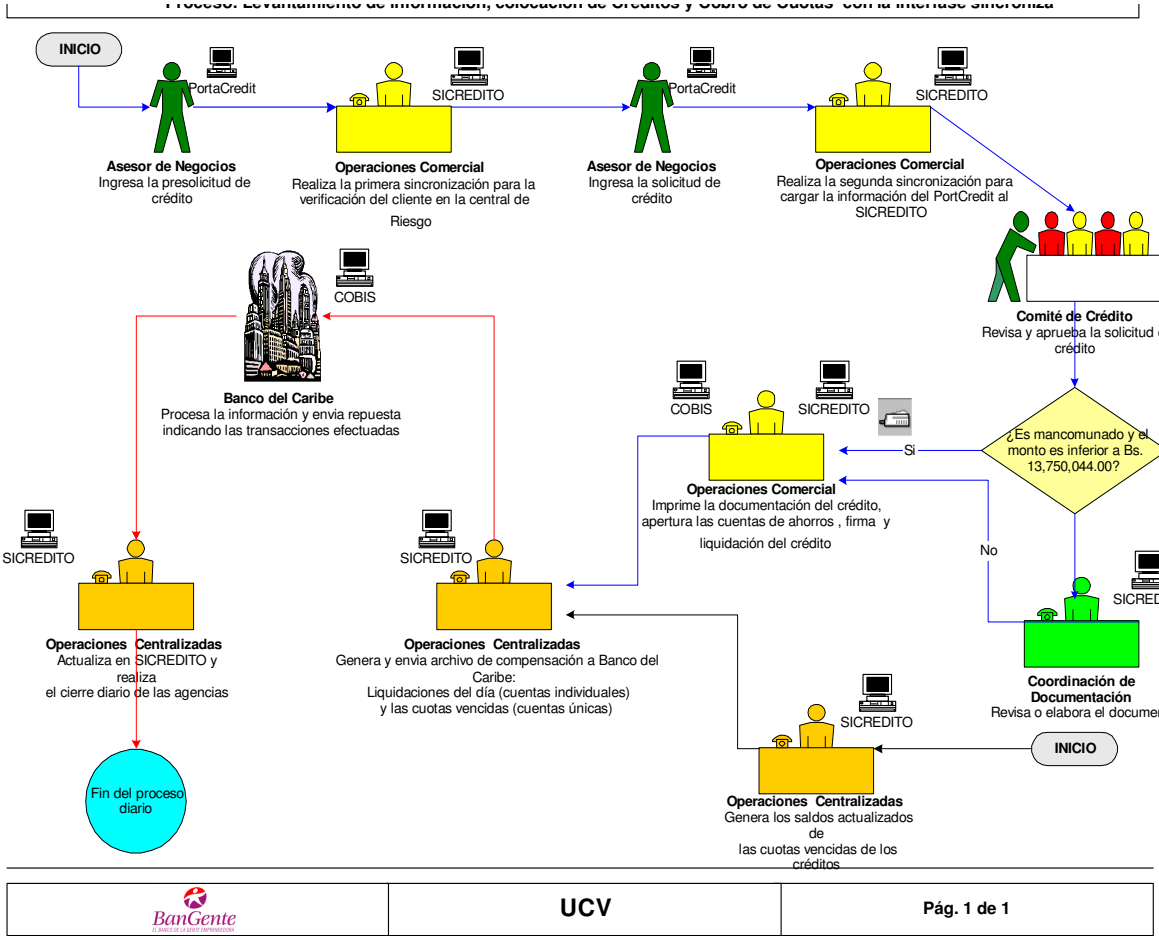


Figura 21. Gráfico de levantamiento de información y colocación de crédito.

Igualmente, de las entrevistas y la observación realizada se puede definir de manera general como va ser el proceso de interfase que apoyara el proceso del modelo operacional declarado anteriormente, en el mismo podemos establecer la relación que va tener la interfase con los distintos módulos del sistema Portacredit y las bases de datos de Sicredito, las cuales serán detalladas mas adelante en la revisión de registros. En la figura 22 podemos detallar con mas exactitud dicha relación.

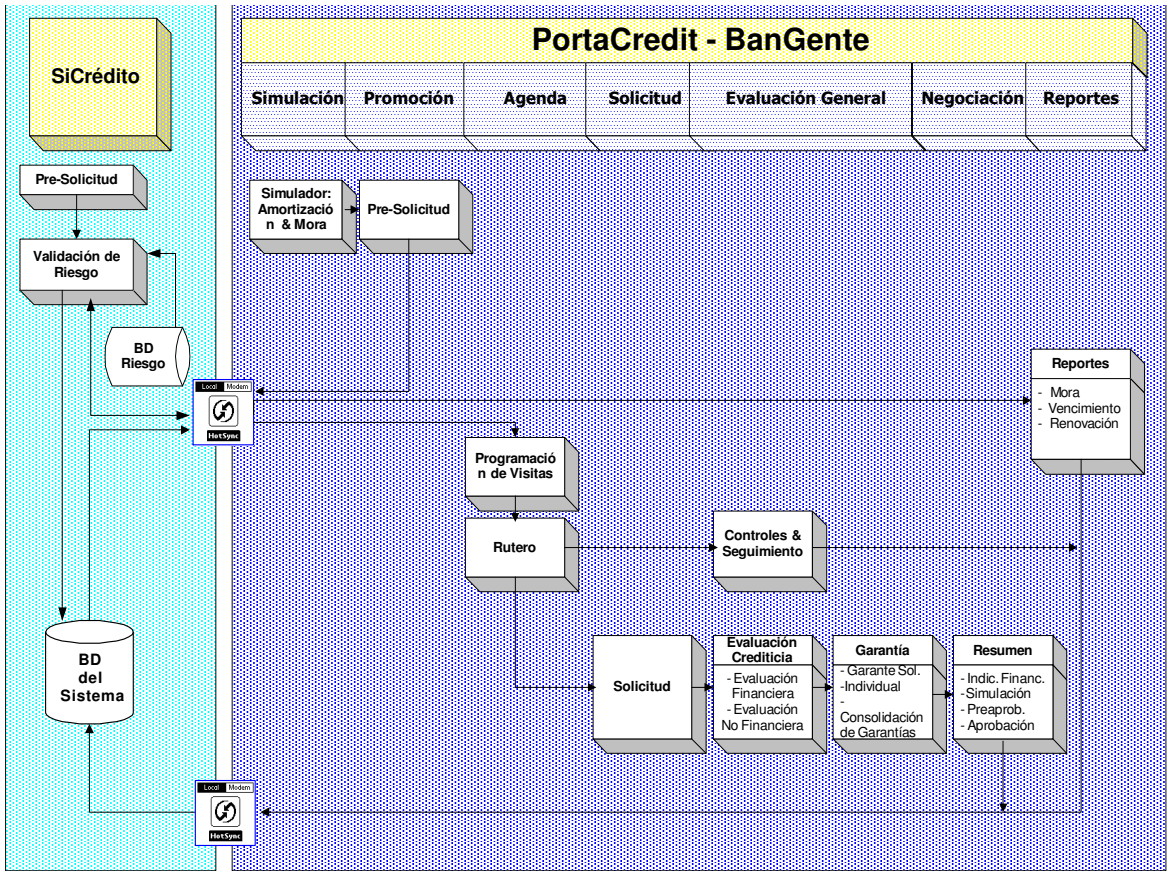


Figura 22. Diagrama general de interfase Portacredit - Sicredito

Relación de registros de datos entre las aplicaciones Portacredit y Sicredito

Mediante la técnica de revisión de registros se establecieron las similitudes entre los campos de las tablas de ambas Bases de Datos, las cuales sirven de referencia para el desarrollo de la interfaz de sincronización; a continuación se visualiza el trabajo realizado de relaciones de datos para las tablas principales del sistema Sicredito (ver tabla 1).

Tabla 1. Relación de datos entre las tablas de los Sistemas Sicredito y Portacredit.

TABLA:	PP_CLIENTE	
Campo Tabla	**Campo PortaCredit**	**Estado en Tabla Palm**
ppcli_id	GCLIENTE.id_cliente	
ppcli_nombres	GCLIENTE.nombre	
ppcli_apellidos	GCLIENTE.apellido	
ppcli_tipo_cliente	'P'	
ppcli_actividad	GCLIENTE.acti_eco	
ppcli_dom_direccion	GDIRECCI.direcci	GDIRECCI.id_lugar = 2
ppcli_dom_telefono	substring(GDIRECCI.telefono,1,14)	GDIRECCI.id_lugar = 2
ppcli_dom_estado	GDIRECCI.estado	GDIRECCI.id_lugar = 2
ppcli_dom_municipio	GDIRECCI.municipio	GDIRECCI.id_lugar = 2
ppcli_neg_direccion	GDIRECCI.direcci	GDIRECCI.id_lugar = 3
ppcli neg telefono	substring(GDIRECCI.telefono,1,14)	GDIRECCI.id_lugar = 3
ppcli neg estado	GDIRECCI.estado	GDIRECCI.id_lugar = 3
ppcli neg municipio	GDIRECCI.municipio	GDIRECCI.id_lugar = 3
ppcli neg parroquia	GDIRECCI.parroq	GDIRECCI.id_lugar = 3
ppcli tiempo experiencia	GCLIENTE.exper_acti	
ppcli referencia ubicación	GDIRECCI.refer	GDIRECCI.id_lugar = 3
ppcli fuente informacion	GCLIENTE.cosupo	
ppcli_asesor	GCLIENTE.id_asesor	
ppcli_nota	GNOTAS.anotacion	
ppcli fecha ult visita	GAGENDA.fecha_in	
ppcli fecha pro visita	GAGENDA.fecha_pro	
ppcli dom parroquia	GDIRECCI.parroq	GDIRECCI.id_lugar = 2
ppcli dom barrio	GDIRECCI.barrio	GDIRECCI.id_lugar = 2
ppcli neg barrio	GDIRECCI.barrio	GDIRECCI.id_lugar = 3
ppcli estado		
ppcli cal int a	Calificacion de riesgo	
ppcli usu cre	GCLIENTE.id_asesor	
ppcli fec cre	getdate()	
ppcli sucursal		
ppcli oficina		
ppcli secuencial		
ppcli fec negacion	Podria venir de Central Riesgo o fec Actual	
ppcli razon negacion	Comentario sobre la negacion	
ppcli cal int b	Validacion Cuentas Vigentes en Riesgo	
ppcli cal int c	Validacion Cuentas Castigadas Riesgo	
ppcli cal int d	Otras Validaciones de Riesgo	

ppcli cal int e	Otras Validaciones de Riesgo	
ppcli cal int f	Otras Validaciones de Riesgo	
ppcli dom sector	GDIRECCI.sector	GDIRECCI.id_lugar = 2
ppcli neg sector	GDIRECCI.sector	GDIRECCI.id_lugar = 3

TABLA:	CR_CAL_INTERNA	
Campo Tabla	**Campo PortaCredit**	**Estado en Tabla Palm**
crcin nacionalidad	GCLIENTE.tipodoc	
crcin cliente	GCLIENTE.id_cliente	
crcin cal a	Calificacion SICRI	
crcin cal b	Validacion Cuentas Vigentes en Riesgo	
crcin cal c	Validacion Cuentas Castigadas en Riesgo	
crcin cal d	Otras Validaciones de Riesgo	
crcin cal e	Otras Validaciones de Riesgo	
crcin cal f	Otras Validaciones de Riesgo	
crcin estado		

TABLA:	CR_CLIENTE	
Campo Tabla	**Campo PortaCredit**	**Estado en Tabla Palm**
crcli id	GCLIENTE.id_cliente	
crcli tipo id	GCLIENTE.tipodoc	
crcli apellidos	GCLIENTE.apellido	
crcli nombres	GCLIENTE.nombre	
crcli nacionalidad	substring(GCLIENTE.nacioncli,1,3)	
crcli nacimiento fecha	GCLIENTE.fecha_nac	Hay que hacer tratamiento Especial
crcli nacimiento lugar	substring(GCLIENTE.lugarnac,1,16)	
crcli sexo	GCLIENTE.sexo	
crcli nivel instrucción	substring(GCLIENTE.gra_instr,1,1)	
crcli estado civil	GCLIENTE.est_civ	Hay que hacer tratamiento Especial
crcli separacion bienes	substring(GCLIENTE.separbien,1,1)	
crcli no cargas	GCLIENTE.numcargas	
crcli dom propio	substring(GCLIENTE.prop_vivi,1,1)	
crcli dom hipoteca	GCLIENTE.prop_vivi	Hay que hacer tratamiento Especial
crcli dom hip fec ven	null	
crcli dom tiempo residencia	isnull(GDIRECCI.tiempo,0)	GDIRECCI.id_lugar = 2
crcli dom direccion	GDIRECCI.direcci	GDIRECCI.id_lugar = 2
crcli dom telefono	substring(GDIRECCI.telefono,1,14)	GDIRECCI.id_lugar = 2
crcli ug org1	GDIRECCI.estado	GDIRECCI.id_lugar = 2
crcli ug org2	GDIRECCI.municipio	GDIRECCI.id_lugar = 2
crcli ug org3	GDIRECCI.parroq	GDIRECCI.id_lugar = 2
crcli ug org4	GDIRECCI.sector	GDIRECCI.id_lugar = 2
crcli ug org5	GDIRECCI.barrio	GDIRECCI.id_lugar = 2
crcli emp nombre	' '	
crcli emp telefono	substring(GDIRECCI.telefono,1,14)	GDIRECCI.id_lugar = 3
crcli ocupacion	GCLIENTE.profesion	
crcli emp cargo ocupa	substring(GCLIENTE.profesion,1,16)	
crcli emp tiempo actual	isnull(GDIRECCI.tiempo,0)	GDIRECCI.id_lugar = 3

crcli emp tiempo anterior	0	
crcli jefe familia	substring(GCLIENTE.jefe_fam,1,1)	
crcli naturaleza	'N'	
crcli exo imp	'N'	
crcli cy id	GCLIENTE.id_conyugue	
crcli cy tipoid	substring(GCLIENTE.tipodocon,1,1)	
crcli cy apellidos	GCLIENTE.apelli_con	
crcli cy nombres	GCLIENTE.nombre_con	
crcli cy emp nombre	GCLIENTE.nomemp_con	
crcli cy emp telefono	substring(GCLIENTE.teleemp_con,1,16)	
crcli cy emp cargo	substring(GCLIENTE.cargo_con,1,16)	
crcli cy emp tiempo	GCLIENTE.antig_con	
crcli fc apellidos	null	
crcli fc nombres	null	
crcli fc parentesco	null	
crcli fc direccion	null	
crcli fc telefono	null	
crcli fc ug org1	null	
crcli fc ug org2	null	
crcli fc ug org3	null	
crcli fc ug org4	null	
crcli fc ug org5	null	
crcli neg lineanegocio	null	
crcli neg localpropio	substring(GCLIENTE.prop_trab,1,1)	
crcli neg direccion	GDIRECCI.direcci	GDIRECCI.id_lugar = 3
crcli neg telefono	substring(GDIRECCI.telefono,1,14)	GDIRECCI.id_lugar = 3
crcli neg ug org1	GDIRECCI.estado	GDIRECCI.id_lugar = 3
crcli neg ug org2	GDIRECCI.municipio	GDIRECCI.id_lugar = 3
crcli neg ug org3	GDIRECCI.parroq	GDIRECCI.id_lugar = 3
crcli neg ug org4	GDIRECCI.sector	GDIRECCI.id_lugar = 3
crcli neg ug org5	GDIRECCI.barrio	GDIRECCI.id_lugar = 3
crcli acc nombres1	null	
crcli acc porcentaje1	null	
crcli acc nombres2	null	
crcli acc porcentaje2	null	
crcli acc nombres3	null	
crcli acc porcentaje3	null	
crcli rep id	null	
crcli rep tipoid	null	
crcli rep apellidos	null	
crcli rep nombres	null	
crcli rep antigüedad	null	
crcli rep fec termino	null	
crcli rep cargo	null	
crcli fec cre	getdate()	
crcli usu cre	GCLIENTE.id_asesor	No creo que se copie en actualizar
crcli fec cam	getdate()	

crcli usu cam	GCLIENTE.id_asesor	
crcli estado	'A'	
crcli sucursal		
crcli oficina		
crcli vinculacion	'N'	
crcli casa prop nombre	substring(GCLIENTE.arr_domi,1,25)	
crcli casa prop telefono	substring(GCLIENTE.tel_arrdomi,1,16)	
crcli emp domicilio	substring(GDIRECCI.direcci,1,30)	GDIRECCI.id_lugar = 3
crcli objetivo social	null	
crcli cy nacionalidad	substring(GCLIENTE.nacion_con,1,3)	
crcli cy nacimiento fecha	GCLIENTE.nacim_con	Hay que hacer tratamiento Especial
crcli cy nacimiento lugar	substring(GCLIENTE.lugnac_con,1,16)	
crcli cy nivel instrucción	substring(GCLIENTE.grinst_con,1,1)	
crcli cy ocupacion	GCLIENTE.cargo_con	
crcli emp direccion	substring(GDIRECCI.direcci,1,80)	
crcli cy emp direccion	GCLIENTE.diremp_con	
crcli dom hipoteca_con	null	
crcli num hijos	0	
crcli num hijos mayores	0	
crcli num hijos estudiantes	0	
crcli num hijos trabajadores	0	
crcli num pers cargo	GCLIENTE.numcargas	
crcli num pers mayores	GCLIENTE.numcargas	
crcli num pers estudiantes	0	
crcli num pers trabajadores	0	
crcli razon social	null	
crcli cuenta	null	
crcli ciudad	'CARACAS'	
crcli croquis direccion	null	
crcli concubino	substring(GCLIENTE.conconcub,1,1)	
crcli cy disponible		Hay que hacer tratamiento Especial
crcli cuanta indiv	null	
crcli voluntad pago	null	
crcli login voluntad pago	null	

TABLA:	CR_ESTADO_FINANCIERO	
Campo Tabla	**Campo PortaCredit**	**Estado en Tabla Palm**
cref cliente	GCLIENTE.id_cliente	
cref usu cre	GCLIENTE.id_asesor	Definir si el cambio es al actualizar
cref fecha cre	getdate()	
cref usu cam	GCLIENTE.id_asesor	
cref fecha cam	getdate()	
cref observaciones	GNOTAS.anotaciones	
cref maq descripcion1	null	
cref maq modelo1	null	
cref maq prendado1	null	

cref maq avaluo1	0.00	
cref maq descripcion2	null	
cref maq modelo2	null	
cref maq prendado2	null	
cref maq avaluo2	0.00	
cref maq descripcion3	null	
cref maq modelo3	null	
cref maq prendado3	null	
cref maq avaluo3	0.00	
cref maq descripcion4	null	
cref maq modelo4	null	
cref maq prendado4	null	
cref maq avaluo4	0.00	
cref inv descripcion1	substring(GCXC_GAR.descrip,1,8)	Primer Registro
cref inv vcmto1	GCXC_GAR.vence	Hay que hacer tratamiento especial
cref inv monto1	GCXC_GAR.monto	
cref inv saldo1	GCXC_GAR.saldo	
cref inv plazo1	convert(int,GCXC_GAR.plazo)	
cref inv descripcion2	substring(GCXC_GAR.descrip,1,8)	Segundo Registro
cref inv vcmto2	GCXC_GAR.vence	Hay que hacer tratamiento especial
cref inv monto2	GCXC_GAR.monto	
cref inv saldo2	GCXC_GAR.saldo	
cref inv plazo2	convert(int,GCXC_GAR.plazo)	
cref inv descripcion3	substring(GCXC_GAR.descrip,1,8)	Tercer Registro
cref inv vcmto3	GCXC_GAR.vence	Hay que hacer tratamiento especial
cref inv monto3	GCXC_GAR.monto	
cref inv saldo3	GCXC_GAR.saldo	
cref inv plazo3	convert(int,GCXC_GAR.plazo)	
cref veh descripcion1	substring(rtrim(GVEHI_GA.marca)+rtrim(GVEHI_GA.linea),1,30)	
cref veh modelo1	GVEHI_GA.ano	Primer Registro
cref veh prendado1	GVEHI_GA.prenda	Hay que hacer tratamiento especial
cref veh avaluo1	GVEHI_GA.avaluo	
cref veh descripcion2	substring(rtrim(GVEHI_GA.marca)+rtrim(GVEHI_GA.linea),1,30)	
cref veh modelo2	GVEHI_GA.ano	Segundo Registro
cref veh prendado2	GVEHI_GA.prenda	Hay que hacer tratamiento especial
cref veh avaluo2	GVEHI_GA.avaluo	
cref veh descripcion3	substring(rtrim(GVEHI_GA.marca)+rtrim(GVEHI_GA.linea),1,30)	
cref veh modelo3	GVEHI_GA.ano	Tercer Registro
cref veh prendado3	GVEHI_GA.prenda	Hay que hacer tratamiento especial
cref veh avaluo3	GVEHI_GA.avaluo	
cref veh descripcion4	substring(rtrim(GVEHI_GA.marca)+rtrim(GVEHI_GA.linea),1,30)	
cref veh modelo4	GVEHI_GA.ano	Cuarto Registro
cref veh prendado4	GVEHI_GA.prenda	Hay que hacer tratamiento especial
cref veh avaluo4	GVEHI_GA.avaluo	
cref pro descripcion1	substring(GINMU_GA.descrip,1,30)	Primer Registro
cref pro area1	convert(int,GINMU_GA.area)	
cref pro direccion1	rtrim(GINMU_GA.ubica)	

cref pro hipotecada1	GINMU_GA.hipoteca	Hay que hacer tratamiento especial
cref pro avaluo1	GINMU_GA.avaluo	
cref pro descripcion2	substring(GINMU_GA.descrip,1,30)	Segundo Registro
cref pro area2	convert(int,GINMU_GA.area)	
cref pro direccion2	rtrim(GINMU_GA.ubica)	
cref pro hipotecada2	GINMU_GA.hipoteca	Hay que hacer tratamiento especial
cref pro avaluo2	GINMU_GA.avaluo	
cref pro descripcion3	substring(GINMU_GA.descrip,1,30)	Tercer Registro
cref pro area3	convert(int,GINMU_GA.area)	
cref pro direccion3	rtrim(GINMU_GA.ubica)	
cref pro hipotecada3	GINMU_GA.hipoteca	Hay que hacer tratamiento especial
cref pro avaluo3	GINMU_GA.avaluo	
cref pre institucion1	GCXP_GAR.institu	Primer Registro
cref pre tipo1	GCXP_GAR.tipocre	Hay que hacer tratamiento especial
cref pre saldo1	GCXP_GAR.saldo	
cref pre cuota1	GCXP_GAR.mensual	
cref pre vcmto1	GCXP_GAR.vence	Hay que hacer tratamiento especial
cref pre plazo1	convert(smallint,GCXP_GAR.plazo)	
cref pre institucion2	GCXP_GAR.institu	Segundo Registro
cref pre tipo2	GCXP_GAR.tipocre	Hay que hacer tratamiento especial
cref pre saldo2	GCXP_GAR.saldo	
cref pre cuota2	GCXP_GAR.mensual	
cref pre vcmto2	GCXP_GAR.vence	Hay que hacer tratamiento especial
cref pre plazo2	convert(smallint,GCXP_GAR.plazo)	
cref pre institucion3	GCXP_GAR.institu	Tercer Registro
cref pre tipo3	GCXP_GAR.tipocre	Hay que hacer tratamiento especial
cref pre saldo3	GCXP_GAR.saldo	
cref pre cuota3	GCXP_GAR.mensual	
cref pre vcmto3	GCXP_GAR.vence	Hay que hacer tratamiento especial
cref pre plazo3	convert(smallint,GCXP_GAR.plazo)	
cref pre institucion4	GCXP_GAR.institu	Cuarto Registro
cref pre tipo4	GCXP_GAR.tipocre	Hay que hacer tratamiento especial
cref pre saldo4	GCXP_GAR.saldo	
cref pre cuota4	GCXP_GAR.mensual	
cref pre vcmto4	GCXP_GAR.vence	Hay que hacer tratamiento especial
cref pre plazo4	convert(smallint,GCXP_GAR.plazo)	
cref rb institucion1	GREFERCL.nombre	tiporef = 'B' and tipocta <> null
cref rb cuenta1	substring(GREFERCL.direccion,1,16)	tiporef = 'B' and tipocta <> null
cref rb tipo1	GREFERCL.tipocta	tiporef = 'B' and tipocta <> null
cref rb observaciones1	GREFERCL.apellido	tiporef = 'B' and tipocta <> null
cref rb institucion2	GREFERCL.nombre	tiporef = 'B' and tipocta <> null
cref rb cuenta2	substring(GREFERCL.direccion,1,16)	tiporef = 'B' and tipocta <> null
cref rb tipo2	GREFERCL.tipocta	tiporef = 'B' and tipocta <> null
cref rb observaciones2	GREFERCL.apellido	tiporef = 'B' and tipocta <> null
cref rb institucion3	GREFERCL.nombre	tiporef = 'B' and tipocta <> null
cref rb cuenta3	substring(GREFERCL.direccion,1,16)	tiporef = 'B' and tipocta <> null
cref rb tipo3	GREFERCL.tipocta	tiporef = 'B' and tipocta <> null

cref rb observaciones3	GREFERCL.apellido	tiporef = 'B' and tipocta <> null
cref tc emisor1	GREFERCL.nombre	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc numero1	substring(GREFERCL.observa,1,16)	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc tipo1	GREFERCL.tipotar	Hay que hacer tratamiento especial
cref tc clase1	GREFERCL.clasetar	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc observaciones1	null	
cref tc emisor2	GREFERCL.nombre	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc numero2	substring(GREFERCL.observa,1,16)	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc tipo2	GREFERCL.tipotar	Hay que hacer tratamiento especial
cref tc clase2	GREFERCL.clasetar	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc observaciones2	null	
cref tc emisor3	GREFERCL.nombre	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc numero3	substring(GREFERCL.observa,1,16)	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc tipo3	GREFERCL.tipotar	Hay que hacer tratamiento especial
cref tc clase3	GREFERCL.clasetar	tiporef = 'B' and tipotar <> null and tipotar <> null
cref tc observaciones3	null	
cref rc institucion1	GREFERCL.nombre	tiporef = 'C'
cref rc sucursal1	GREFERCL.apellido	tiporef = 'C'
cref rc telefono1	substring(GREFERCL.telefono,1,16)	tiporef = 'C'
cref rc fecha1	GREFERCL.fechacre	Hay que hacer tratamiento especial
cref rc monto1	GREFERCL.valor	tiporef = 'C'
cref rc forma pago1	GREFERCL.tipocta	tiporef = 'C'
cref rc observaciones1	null	
cref rc institucion2	GREFERCL.nombre	tiporef = 'C'
cref rc sucursal2	GREFERCL.apellido	tiporef = 'C'
cref rc telefono2	substring(GREFERCL.telefono,1,16)	tiporef = 'C'
cref rc fecha2	GREFERCL.fechacre	Hay que hacer tratamiento especial
cref rc monto2	GREFERCL.valor	tiporef = 'C'
cref rc forma pago2	GREFERCL.tipocta	tiporef = 'C'
cref rc observaciones2	null	
cref rc institucion3	GREFERCL.nombre	tiporef = 'C'
cref rc sucursal3	GREFERCL.apellido	tiporef = 'C'
cref rc telefono3	substring(GREFERCL.telefono,1,16)	tiporef = 'C'
cref rc fecha3	GREFERCL.fechacre	Hay que hacer tratamiento especial
cref rc monto3	GREFERCL.valor	tiporef = 'C'
cref rc forma pago3	GREFERCL.tipocta	tiporef = 'C'
cref rc observaciones3	null	
cref ac efectivo (1)	isnull(GEVA_FAM.ac_bancos,0.00)	
cref ac cxc (2)	isnull(GEVA_FAM.ac_cxc, 0.00)	
cref ac inventario	0.00	
cref ac total	(1) + (2)	
cref af muebles (3)	isnull(GEVA_FAM.ac_muebles, 0.00)	
cref af vehiculos (4)	isnull(GEVA_FAM.ac_vehiculo, 0.00)	
cref af inmuebles (5)	isnull(GEVA_FAM.ac_inmuebl, 0.00)	
cref af otros monto (6)	isnull(GEVA_FAM.ac_otracti, 0.00)	
cref af total	(3) + (4) + (5) + (6)	
cref activos total (7)	isnull(GEVA_FAM.tot_activo, 0.00)	

cref pas prestamos per	isnull(GEVA_FAM.pa_prueban, 0.00)	
cref pas hipotecas	0.00	
cref pas otros monto	isnull(GEVA_FAM.pa_otros, 0.00)	
cref pas total (8)	isnull(GEVA_FAM.tot_pasivo, 0.00)	
cref pas cxp	isnull(GEVA_FAM.pa_cxp, 0.00	
cref pas efectos x p	isnull(GEVA_FAM.pa_efectos, 0.00)	
cref pas impuesto	isnull(GEVA_FAM.pa_impuest, 0.00)	
cref capacidad pago	0.00	
cref patrimonio	(7) - (8)	
cref ing monto	isnull(GEVA_FAM.in_salario, 0.00)	
cref ing monto conyugue	isnull(GEVA_FAM.in_conyugue, 0.00)	
cref ing rendimiento	isnull(GEVA_FAM.in_interes,0.00)	
cref ing otros1	null	
cref ing otros monto1	isnull(GEVA_FAM.in_otros, 0.00)	
cref ing otros2	null	
cref ing otros monto2	0.00	
cref ing otros3	null	
cref ing otros monto3	0.00	
cref ing total	isnull(GEVA_FAM.tot_ingres, 0.00)	
cref egr arriendo	isnull(GEVA_FAM.ga_vivien, 0.00)	
cref egr servicios	isnull(GEVA_FAM.ga_servici, 0.00)	
cref egr hogar	isnull(GEVA_FAM.ga_educac, 0.00)	
cref egr tc	isnull(GEVA_FAM.ga_vestido, 0.00)	
cref egr pre	isnull(GEVA_FAM.ga_alimen, 0.00)	
cref egr otros	'otros'	
cref egr otros monto	isnull(GEVA_FAM.ga_otros, 0.00)	
cref egr total	isnull(GEVA_FAM.tot_gasts, 0.00)	
cref tipo	'N'	
cref ahorro mensual	isnull(GEVA_FAM.ga_ahorro, 0.00)	
cref fecha	getdate()	
cref pas deudas 180	isnull(GEVA_FAM.pa_men180, 0.00)	
cref pas deudas m180	isnull(GEVA_FAM.pa_mas180, 0.00)	
cref rf nombre1	GREFERCL.apellido + GREGERCL.nombre	GREFERCL.tiporef = 'F'
cref rf nexo1	substring(GREFERCL.tipocta,1,4)	GREFERCL.tiporef = 'F'
cref rf telefono1	substring(GREFERCL.telefono,1,15)	GREFERCL.tiporef = 'F'
cref rcom nombre1	GREFERCL.apellido + GREGERCL.nombre	GREFERCL.tiporef = 'V'
cref rcom antiguedad1	convert(varchar,GREFERCL.plazo)	GREFERCL.tiporef = 'V'
cref rcom referencia1	GREFERCL.tipotar	GREFERCL.tiporef = 'V'
cref rf nombre2	GREFERCL.apellido + GREGERCL.nombre	GREFERCL.tiporef = 'F'
cref rf nexo2	substring(GREFERCL.tipocta,1,4)	GREFERCL.tiporef = 'F'
cref rf telefono2	substring(GREFERCL.telefono,1,15)	GREFERCL.tiporef = 'F'
cref rcom nombre2	GREFERCL.apellido + GREGERCL.nombre	GREFERCL.tiporef = 'V'
cref rcom antiguedad2	convert(varchar,GREFERCL.plazo)	GREFERCL.tiporef = 'V'
cref rcom referencia2	GREFERCL.tipotar	GREFERCL.tiporef = 'V'
cref rf nombre3	GREFERCL.apellido + GREGERCL.nombre	GREFERCL.tiporef = 'F'
cref rf nexo3	substring(GREFERCL.tipocta,1,4)	GREFERCL.tiporef = 'F'
cref rf telefono3	substring(GREFERCL.telefono,1,15)	GREFERCL.tiporef = 'F'

cref rcom nombre3	GREFERCL.apellido + GREGERCL.nombre	GREFERCL.tiporef = 'V'
cref rcom antiguedad3	convert(varchar,GREFERCL.plazo)	GREFERCL.tiporef = 'V'
cref rcom referencia3	GREFERCL.tipotar	GREFERCL.tiporef = 'V'
cref pas otras obligaciones	isnull(GEVA_FAM.pa_otrobli, 0.00)	
cref egr condominio	isnull(GEVA_FAM.ga_condom, 0.00)	
cref egr salud	isnull(GEVA_FAM.ga_salud	
cref rcom telefono1	substring(GREFERCL.telefono,1,15)	GREFERCL.tiporef = 'V'
cref rcom telefono2	substring(GREFERCL.telefono,1,15)	GREFERCL.tiporef = 'V'
cref rcom telefono3	substring(GREFERCL.telefono,1,15)	GREFERCL.tiporef = 'V'
cref rcom direccion1	rtrim(GREFERCL.direccion)	GREFERCL.tiporef = 'V'
cref rcom direccion2	rtrim(GREFERCL.direccion)	GREFERCL.tiporef = 'V'
cref rcom direccion3	rtrim(GREFERCL.direccion)	GREFERCL.tiporef = 'V'
cref rf direccion1	GREFERCL.direccion	GREFERCL.tiporef = 'F'
cref rf direccion2	GREFERCL.direccion	GREFERCL.tiporef = 'F'
cref rf direccion3	GREFERCL.direccion	GREFERCL.tiporef = 'F'
cref unidad expresion	null	

TABLA:	CR_MICROEMPRESA	
Campo Tabla	**Campo PortaCredit**	**Estado en Tabla Palm**
crmic id		Se tiene que calcular en el sistema
crmic rif	GCLIENTE.id_cliente	
crmic nombre	'NO TIENE'	
crmic razon social	'O'	
crmic nombre socio	'NO TIENE'	
crmic sector	GCLIENTE.sectoreco	
crmic experiencia	GCLIENTE.exper_acti	
crmic_nivel_tecnif	null	
crmic_empl_familiares	0	
crmic_empl_externos	0	
crmic_promedio_ventas	GCLIENTE.aux1	
crmic_promedio_up	'M'	
crmic_hipoteca	'N'	
crmic_hipoteca_con	null	
crmic_arrendador	GCLIENTE.arr_local	
crmic_dom_antiguedad	GCLIENTE.anti_nego	
crmic_ciiu	GCLIENTE.acti_eco	
crmic_actividad	'ACTIVIDAD NO BIEN ESPECI'	
crmic_pri_producto1	'VARIOS'	
crmic_pri_producto2	null	
crmic_pri_producto3	null	
crmic_num_trabajadores	0	
crmic_num_horas	0	
crmic_tiempo_experiencia	GCLIENTE.exper_acti	
crmic_local_propio	SUBSTRING(GCLIENTE.prop_trab,1,1)	
crmic_tiempo_local	isnull(GDIRECCI.tiempo,0)	GDIRECCI.id_lugar = 3
crmic_num_empleos	null	
crmic_capacitacion_interna	null	

crmic_lugar_ventas	null	
crmic_direccion	substring(GDIRECCI.direcci,1,60)	GDIRECCI.id_lugar = 3
crmic_telefono	substring(GDIRECCI.telefono,1,16)	GDIRECCI.id_lugar = 3
crmic_ug_org1	GDIRECCI.estado	GDIRECCI.id_lugar = 3
crmic_ug_org2	GDIRECCI.municipio	GDIRECCI.id_lugar = 3
crmic_ug_org3	GDIRECCI.parroq	GDIRECCI.id_lugar = 3
crmic_ug_org4	GDIRECCI.sector	GDIRECCI.id_lugar = 3
crmic_ug_org5	GDIRECCI.barrio	GDIRECCI.id_lugar = 3
crmic_cuci	null	
crmic_ac_efectivo	isnull(GDAT_FIN.totdisponi, 0.00) (1)	
crmic_ac_cxc	isnull(GDAT_FIN.totctaxcob, 0.00) (2)	
crmic_ac_materia_prima	isnull(GDAT_FIN.matprimas, 0.00) (3)	
crmic ac mercaderia insumos	isnull(GDAT_FIN.enventa, 0.00) (4)	
crmic ac productos terminados	isnull(GDAT_FIN.terminados, 0.00) (5)	
crmic ac productos proceso	isnull(GDAT_FIN.enproceso, 0.00) (6)	
crmic_ac_total	(1) + (2) + (3) + (4) + (5) + (6)	
crmic_af_otros_monto (7)	isnull(otrosactiv,0) + isnull(totmaqequi,0)	
crmic_af_activos_fijos (8)	isnull(GDAT_FIN.totactinmu,0)	
crmic_af_total	(7) + (8)	
crmic_act_total (9)	isnull(GDAT_FIN.a_total, 0.00)	
crmic pas deudas corto plazo	isnull(GDAT_FIN.deuda180, 0.00)	
crmic pas deudas largo plazo	isnull(GDAT_FIN.p_cxplargo, 0.00)	
crmic pas deudas m 180	isnull(GDAT_FINdeuda200, 0.00)	
crmic_pas_cxp	isnull(pres_gar,0) + isnull(otras_obli,0)	
crmic_pas_exp	isnull(GDAT_FIN.efxpagar, 0.00)	
crmic_pas_prestamos_b	isnull(GDAT_FIN.p_presbalp, 0.00)	
crmic_pas_impuestos	0	
crmic_pas_hipotecas	isnull(GDAT_FIN.p_hipote, 0.00)	
crmic_pas_otros_pasivos	isnull(GDAT_FIN.p_otrospas, 0.00)	
crmic_pas_total (10)	isnull(GDAT_FIN.p_total, 0.00)	
crmic_pas_capital	(9) - (10)	
crmic_ecf_remuneracion	isnull(SUM(GREPERSO.valtotal), 0.00)	GREPERSO.tiporemu = '1'
crmic_ecf_mano_obra	isnull(SUM(GREPERSO.valtotal), 0.00)	GREPERSO.tiporemu = '2'
crmic_ecv_mano_obra	isnull(SUM(GREPERSO.valtotal), 0.00)	GREPERSO.tiporemu in ('3','4','5')
crmic_ecf_comisiones	0	
crmic_ecv_comisiones	0	
crmic_ecv_materias_primas	isnull(GDAT_FIN.costovta, 0.00)	
crmic_ecv_mercaderias	isnull(GDAT_FIN.transpor, 0.00)	
crmic_ecf_combustibles	isnull(GDAT_FIN.agualuzloc, 0.00)	
crmic_ecv_combustibles	0	
crmic_ecf_mantenimiento	isnull(GDAT_FIN.otrospagos, 0.00)	
crmic_ecv_servicios_terceros	isnull(GDAT_FIN.imprevis,0)	
crmic_ecf_alquiler_local	isnull(GDAT_FIN.alquilerlo,0)	
crmic_ecf_gastos_financieros	isnull(GDAT_FIN.gtosfinan,0)	
crmic_ecf_impuestos	isnull(GDAT_FIN.impuestos, 0.00)	
crmic_ecv_impuestos	0	
crmic_ecf_seguro_social	0	

crmic_ecv_seguro_social	0	
crmic_ecf_depreciaciones	0	
crmic_ecf_otros_gastos	isnull(GEVA_FAM.tot_gastos, 0.00)	Hay que hacer calculo especial
crmic_ecv_otros_gastos	0	
crmic_ecf_total	0	Se calcula en el sistema
crmic_ecv_total	0	Se calcula en el sistema
crmic pyg ventas mensuales	isnull(GDAT_FIN.ventasmes, 0.00)	
crmic_pyg_otros_ingresos	isnull(GEVA_FAM.tot_ingres, 0.00)	Hay que hacer calculo especial
crmic_pyg_costos_totales		
crmic_pyg_gastos_hogar		
crmic_pyg_perdida_ganancia		
crmic_ia_num_proveedores		
crmic_ia_cyt_impresion		
crmic_ia_cyt_transparencia		
crmic_ia_cyt_accesibilidad		
crmic_ia_cyt_entorno		
crmic_ia_cge_experiencia		
crmic_ia_cge_conocimiento		
crmic_ia_cge_iniciativa		
crmic_ia_cge_claridad		
crmic_ia_num_clientes		
crmic_ia_vx_menor		
crmic ia vxme zona comercial		
crmic ia vxme buena ubicacion		
crmic ia vxme competencia oyd		
crmic_ia_vxme_anuncios		
crmic_ia_vxme_afluencia		
crmic_ia_vx_mayor		
crmic_ia_vxma_dependencia		
crmic_ia_vxma_pedido		
crmic_ia_vxma_exporta		
crmic_cliente	GCLIENTE.id_cliente	
crmic_acf_nombre1	SUBSTRING(GACT_INM.direccion,1,30)	Primer Registro
crmic_acf_marca1	SUBSTRING(GACT_INM.escritura,1,10)	
crmic_acf_valor1	GACT_INM.valor	
crmic_acf_nombre2	SUBSTRING(GACT_INM.direccion,1,30)	Segundo Registro
crmic_acf_marca2	SUBSTRING(GACT_INM.escritura,1,10)	
crmic_acf_valor2	GACT_INM.valor	
crmic_acf_nombre3	SUBSTRING(GACT_INM.direccion,1,30)	Tercer Registro
crmic_acf_marca3	SUBSTRING(GACT_INM.escritura,1,10)	
crmic_acf_valor3	GACT_INM.valor	
crmic_cxp_acreedor1	'Prestamos Bancarios a CP'	
crmic_cxp_plazo1	null	
crmic_cxp_saldo1	isnull(GDAT_FIN.pres_gar,0)	
crmic_cxp_antiguedad1	null	
crmic_cxp_acreedor2	'Otras Obligaciones a CP'	
crmic_cxp_plazo2	null	

crmic_cxp_saldo2	isnull(GDAT_FIN.otras_obli,0)	
crmic_cxp_antiguedad2	null	
crmic_cxp_acreedor3	null	
crmic_cxp_plazo3	null	
crmic_cxp_saldo3	0	
crmic_cxp_antiguedad3	null	
crmic_inv_descripcion1	'Cuentas por Cobrar Totales'	
crmic_inv_vcmto1	null	
crmic_inv_monto1	isnull(GDAT_FIN.cc_totales, 0.00)	
crmic_inv_saldo1	isnull(GDAT_FIN.cc_totales, 0.00)	
crmic_inv_plazo1	null	
crmic_inv_descripcion2	'Por Cobrar de Dudosa Recupera'	
crmic_inv_vcmto2	null	
crmic_inv_monto2	isnull(-1 * GDAT_FIN.cc_dudosa, 0.00)	
crmic_inv_saldo2	isnull(-1 * GDAT_FIN.cc_dudosa, 0.00)	
crmic_inv_plazo2	null	
crmic_inv_descripcion3	null	
crmic_inv_vcmto3	null	
crmic_inv_monto3	0	
crmic_inv_saldo3	0	
crmic_inv_plazo3	null	
crmic_af_maquinaria	isnull(GDAT_FIN.totmaqequi, 0.00)	
crmic_pas_otras_obli	isnull(GDAT_FIN.p_otroslp, 0.00)	
crmic_pas_superavit	isnull(GDAT_FIN.res_super, 0.00)	
cmic_org_nego_1		
cmic_org_nego_2		
cmic_org_nego_3		
cmic_org_nego_4		
cmic_tran_inf_1		
cmic_tran_inf_2		
cmic_tran_inf_3		
cmic_tran_inf_4		

TABLA:	CR_SOLICITUDES	
Campo Tabla	**Campo PortaCredit**	**Estado en Tabla Palm**
crsol_solicitud		
crsol_tipo		
crsol_sucursal		
crsol_oficina		
crsol_fecha	GETDATE()	
crsol_cliente	GCLIENTE.id_cliente	
crsol_oficial	GSOLICIT.id_asesor	
crsol_agente	'1'	
crsol_usu_cre		Hay que ver, tiene que ser usuario operaciones
crsol_fec_cre	GETDATE()	
crsol_tipo_credito	'CD'	
crsol_destino_prestamo	'COMPRA DE MATERIA PRIMA'	

crsol_monto	GSOLICIT.montosol	
crsol_moneda	'S'	
crsol_plazo	GSOLICIT.plazosol*30	
crsol_forma_pago	GSOLICIT.ppago	Hay que hacer tratamiento especial
crsol_microempresa		Este es el id de microempresa que se calcula
crsol_observaciones	null	
crsol_estado	'I'	
crsol_renovado_de		
crsol_renovado_en		
crsol_destino_fondos	'CAPTRA'	
crsol_map_producto	GSOLICIT.producto	Hay que hacer tratamiento especial
crsol_map_nivel		
crsol_map_fecha1		
crsol_map_usuario1		
crsol_map_estado1		
crsol_map_anotacion1		
crsol_map_fecha2		
crsol_map_usuario2		
crsol_map_estado2		
crsol_map_anotacion2		
crsol_map_riesgo		
crsol_map_cancelacion		
crsol_map_tipo_tabla		
crsol_map_fuente_recursos		
crsol_map_reajuste		
crsol_map_instrucciones		
crsol_map_monto		
crsol_map_plazo		
crsol_map_tasa		
crsol_map_forma_pago		
crsol_map_gracia		
crsol_pi_monto_capital	GSOLICIT.montosol	
crsol_pi_monto_adq_equipo	0	
crsol_pi_monto_adq_edificios	0	
crsol_pi_monto_adq_muebles	0	
crsol_pi_otros	0	
crsol_pi_saldo_op_anterior	0	
crsol_pi_monto_solicitado	GSOLICIT.montosol	
crsol_sol_grupo	GGRUPO.nombre	Se aplicara tratamiento de grupo o individual
crsol_sol_participantes	GGRUPO.numinte	Se aplicara tratamiento de grupo o individual
crsol_desembolso_tipo_cta		
crsol_uso_cantidad1		
crsol_uso_descripcion1		
crsol_conting_tasa		
crsol_conting_comision		
crsol_uso_unidad_medida2		
crsol_uso_cantidad2		

crsol_uso_descripcion2		
crsol_uso_precio_unitario2		
crsol_uso_precio_total2		
crsol_uso_unidad_medida3		
crsol_uso_cantidad3		
crsol_uso_descripcion3		
crsol_uso_precio_unitario3		
crsol_uso_precio_total3		
crsol_uso_unidad_medida4		
crsol_uso_cantidad4		
crsol_uso_descripcion4		
crsol_uso_precio_unitario4		
crsol_uso_precio_total4		
crsol_uso_unidad_medida1		
crsol_uso_cantidad5		
crsol_uso_descripcion5		
crsol_uso_precio_unitario5		
crsol_uso_precio_total5		
crsol_uso_unidad_medida6		
crsol_uso_cantidad6		
crsol_uso_descripcion6		
crsol_uso_precio_unitario6		
crsol_uso_precio_total6		
crsol_calificacion	'S' o 'N'	Depende de si es individual a grupo
crsol_map_tasa_activa		
crsol_impresiones		
crsol_fecha_otorgamiento		
crsol_map_cotizacion		
crsol_num_pagos		
crsol_tipo_garantia		
crsol_total_a_recibir		
crsol_operacion		
crsol_usu_otorgamiento		
crsol_map_num_cuotas		
crsol_oro_peso_sobre		
crsol_oro_peso_oro		
crsol_oro_total_avaluo		
crsol_oro_cupo_credito		
crsol_oro_custodio		
crsol_novedad		
crsol_novedades		
crsol_renovacion_no		
crsol_cuenta		
crsol_tipo_desembolso		
crsol_fecha_desembolso		
crsol_linea_credito		
crsol_no_beneficiados		

crsol_conting_beneficiario		
crsol_conting_garantizar		
crsol_uso_precio_unitario1		
crsol_uso_precio_total1		
crsol_desembolso_ref		
crsol_desembolso_banco		
crsol_factoring_tipo_tasa		
crsol_puntos_reajuste		
crsol_uso_unidad_medida5		
crsol_map_usuario3		
crsol_map_estado3		
crsol_map_anotacion3		
crsol_map_usuario4		
crsol_map_estado4		
crsol_map_anotacion4		
crsol_map_fecha3		
crsol_map_fecha4		
crsol cancelacions en proceso		
crsol_map_tipo_tasa		
crsol_num_empleos		
crsol_capacidad_pago		
crsol_map_tasa_calculo		
crsol_usu_consultor		
crsol_revisado		
crsol_porcentaje_serfin		
crsol_gracia_reajuste		
crsol porcentaje SNF pagado		
crsol_monto_SNF_pagado		
crsol_descuento_SNF		
crsol_map_tasa_efectiva		
crsol_liquida_cta_manc		

RESULTADOS

Ante todo hay que destacar que se logro una solución computacional que aprovecha la tecnología de las Palm Pilot para automatizar los procesos de captura de información de clientes y seguimiento de la cartera de crédito del asesor de negocio, logrando así que los objetivos y alcances de este trabajo de grado se cubrieran con éxito.

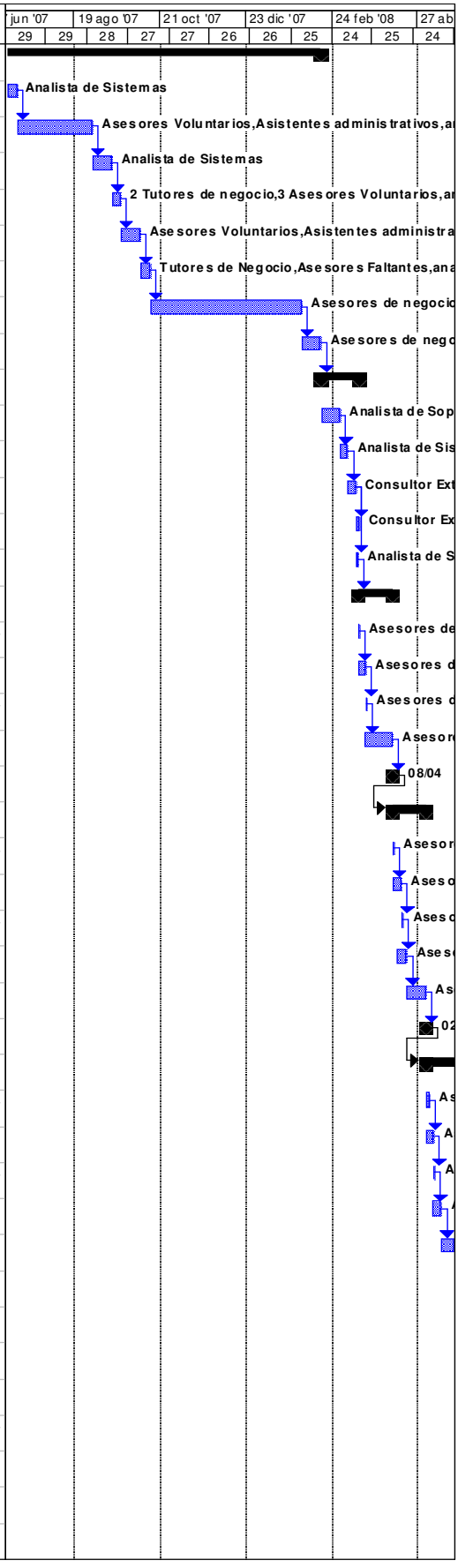
Durante el proceso de desarrollo, pruebas, fase piloto e implantación se evidencio la potencialidad de todo el proceso para aumentar la productividad de los asesores de negocio y las asistentes administrativas, ya que permitió agilizar los trámites de crédito, brindando un mejor servicio al cliente al evitar que los asesores de negocios tengan que llevar esta información de forma manual para su posterior trascripción; de esta manera se apoyo a la organización a minimizar el consumo de papel controlando los costos y apoyando a la conservación del medio ambiente.

Un tema importante que debemos señalar es el presentado con el proceso de desarrollo de la interfase "sincroniza", el cual debió ser desarrollado bajo la plataforma de programación Power Builder 9.0 exigida por la gerencia de tecnología de la organización BanGente, en el cual, por problemas acontecidos con el archivo active .ocx que mantiene la comunicación y control de la sincronización entre la Palm y el PC, bajo la plataforma prevista no se podía establecer control sobre dicho conductor de datos, situación validada con la compañía creadora de dicha plataforma (Sybase) informando que se encuentra en proceso de estudio el caso de error, por lo cual se tuvo que optar por la opción de desarrollar bajo la plataforma de Power Builder 6.5.

Cronograma de implantación para fase piloto y fase inicial de masificación

En el siguiente cronograma se visualizan las fases de prueba que se tuvieron que seguir para cumplir con los procesos de certificación de la piloto y su correspondiente certificación de la masificación en agencia.

Id	Nombre de tarea	Duración
1	**Fase Piloto Agencia Catia**	**165 días**
2	Adiestramiento 2 asesores voluntarios	5 días
3	Fase de prueba Aplicaciones paralela para 2 asesores voluntarios	40 días
4	Acondicionamiento de interfase y aplicación Portacredit por resultados de las pruebas pilotos	10 días
5	Adiestramiento 3 asesores voluntariuos nuevos	5 días
6	Fase de prueba adicional aplicaciones paralelo para 5 asesores voluntarios	10 días
7	Masificación y adiestramiento asesores faltantes de agencia piloto	5 días
8	Piloto con Bases de datos en producción, incluye acondicionamiento de interfase y aplicación.	80 días
9	Fase final de piloto	10 días
10	**Preparación Masificación**	**20 días**
11	Adquisición dispositivos PalmPilot	10 días
12	Configuración de los dispositivos	4 días
13	Adiestramiento Básico (Manejo dispositivo + aplicación PortaCredit)	3 días
14	Adiestramiento Personal de Sistemas	3 días
15	Elaboración de ejercicios prácticos	2 días
16	**Masificación Agencia Centro**	**18 días**
17	Adiestramiento en Agencia - Ingreso de casos de prueba en Porta Credit	1 día
18	Ingreso de casos propios en Porta Credit - Sincronización en base de datos de prueba en agencia	4 días
19	Adiestramiento en Agencia - Sesión de preguntas y respuestas de los problemas presentados	1 día
20	Ingreso de casos reales en Portacredit, Sincronización en producción y seguimiento de la actividad	15 días
21	**Certificación Agencia**	**0 días**
22	**Masificación Agencia Palo Verde**	**18 días**
23	Adiestramiento en Agencia - Ingreso de casos de prueba en Porta Credit	1 día
24	Ingreso de casos propios en Porta Credit - Sincronización en base de datos de prueba en agencia	4 días
25	Adiestramiento en Agencia - Sesión de preguntas y respuestas de los problemas presentados	1 día
26	Ingreso de casos reales en Portacredit	5 días
27	Revisión de casos ingresado, Sincronización en producción y seguimiento de la actividad	10 días
28	Certificación Agencia	0 días
29	**Masificación Agencia El Valle**	**18 días**
30	Adiestramiento en Agencia - Ingreso de casos de prueba en Porta Credit	1 día
31	Ingreso de casos propios en Porta Credit - Sincronización en base de datos de prueba en agencia	4 días
32	Adiestramiento en Agencia - Sesión de preguntas y respuestas de los problemas presentados	1 día
33	Ingreso de casos reales en Portacredit	5 días
34	Revisión de casos ingresado, Sincronización en producción y seguimiento de la actividad	10 días
35	Certificación Agencia	0 días
36	**Masificación Agencia Valencia Michelena**	**21 días**
37	Adiestramiento Intensivo en Agencia	2 días
38	Ingreso de casos propios en Porta Credit - Sincronización en base de datos de prueba en agencia	3 días
39	Adiestramiento en Agencia - Sesión de preguntas y respuestas de los problemas presentados	1 día
40	Ingreso de casos reales en Portacredit	5 días
41	Revisión de casos ingresado, Sincronización en producción y seguimiento de la actividad	10 días
42	Certificación Agencia	0 días



CONCLUSIONES Y RECOMENDACIONES

Una vez concluido el análisis, diseño, desarrollo e implantación del sistema basado en tecnología Palm Pilot y su interfaz con el sistema industrial de crédito de la organización, podemos concluir lo siguiente:

- Una vez llevado a cabo el rediseño, mejoramiento, la automatización de los procesos y la puesta en marcha del sistema, se solucionaron problemas derivados por errores en los datos de las Bases de Datos que soportaban las solicitudes de crédito involucrados en el proceso de transcripción de información de los asistentes administrativo y que ahora son controladas desde la aplicación Portacredit, siendo trasladados confiablemente a través de la interfaz hasta el sistema Sicredito.

- Con la puesta en marcha de la automatización de los procesos de levantamiento de información de solicitudes de crédito se obtuvieron cambios en beneficio del desarrollo y del buen funcionamiento de la organización, entre las cuales se destaca:
    - Disminución de los tiempos de respuesta a los clientes.
    - Aumento de la productividad del personal involucrado en los procesos.
    - Reducción en los costos involucrados en los procesos atacados, sobre todo los costos de papelería.
    - Incremento en la calidad del servicio prestado.

- La utilización del Método Integrado Orientado a Objeto con el rediseño de procesos, sirvió para identificar los requerimientos del sistema, rediseñar los procesos y diseñar la aplicación. Permitiendo así, identificar las funcionalidades del sistema expresados como casos de uso a partir del análisis de los

requerimientos y las entrevistas a los usuarios, para la construcción del sistema, mejorando así la eficiencia y la eficacia de los procesos.

- Se lograron los objetivos propuestos con respecto a la integración automática de los procesos manuales del personal de asesores de negocio y asistentes administrativos de la organización BanGente.

En este trabajo se platean las siguientes recomendaciones:

- Se recomienda continuar afianzando y explotando la utilización de la plataforma de desarrollo PowerBuilder para poder realizar la migración de la interfaz desarrollada a versiones posteriores a la de PowerBuilder 6.5.

- Para obtener una solución total y aun más eficiente, se debe trabajar en un mecanismo de intercambios de información de la interfaz vía microondas, WEB o satelital, así como también la implantación de una interfaces multiusuario, sobre la cual puedan sincronizar un amplio grupo de asesores de negocios paralelamente.

- Se recomienda trabajar en la implantación de agendas compartidas y la posibilidad de espacios compartidos virtuales, donde ambos, clientes y técnicos puedan observar el estado de los procesos.

REFERENCIAS BIBLIOGRÁFICAS

[ACTIV] ACTIV@MENTE. *VRML – Realidad Virtual*. Pagina Web. Visitada 21/05/2008. http://www.activamente.com.mx/vrml/aplicaciones.html

[VALZA] VALZACCHI. *Mundos Virtuales*. Pagina Web. Visitada 25/05/2008. http://www.educoas.org/portal/bdigital/contenido/valzacchi/valzacchiCapitulo-13New.pdf

[GON00] SOSA GONZALEZ, JORGE E. *La Reingeniería para mejorar la Rentabilidad de la Empresa*. Pagina Web. Visitada 31/04/2008. http://www.geocities.com/estudioeic/reingenieria.htm

[INFOR] *Informática y Reingeniería*. Pagina Web. Visitada 11/07/2002. http://www.inei.gob.pe/cpi/bancopub/libfree/lib659/CAP2.htm

[KEN91] KENDALL Y KENDALL. *Análisis y Diseño de Sistemas*. Editorial Prentice may. México 1991.

[MAS96] MAS, JORDI. *Mundos Virtuales en Internet.* Pagina Web. 1996. Visitada 21/10/2002. http://www.lleida.net/clientes/jmas/vrml.htm

[OLI92] OLIVETTI. *Diccionario de Informática*. Editorial Paraninfo; España-Madrid 1992.

[PALM99] "*Seams Teach Yourself Palm Programming in 24 Hours*". Editorial Sams Publishing; USA 1999.

[SEN92] SEEN, JAMES A. *Análisis y Diseño de Sistemas de Información*. Editorial Mc. Graw Hill; España 1992.

[UML04] http://www.monografias.com/new/2004-09-19.shtml → links/computación Añadido: dom sep 19 2004 trabajo → UML → http://usuarios.lycos.es/oopere/uml.htm.

[JAC96] Jacobson, I. "Object Oriented Software Engineering: A use case driven approach", Addison – Wesley Publishing Company. 1996